1. 基本概念篇

1.1 什么是 Sybase Adaptive Server Enterprise （ASE）？

Sybase Adaptive Enterprise Server （ASE）是由Sybase公司提供的具有高性能、高可靠性和易维护性的一个关系数据库管理系统。

1.2 Sybase 具有哪些 Server以及这些 Server的用途？

Adaptive Server

Adaptive Server是Sybase公司提供的适应性核心数据库服务器，用于管理整个数据库系统，包括用户、数据、资源等的管理和控制。

Backup Server

在备份和恢复时执行数据库的 dump 和 load.

XP Server

执行扩展存储过程

Monitor Server

为性能调试分析采集数据

Historical Server

保存来自Monitor Server 的数据,以备将来分析.

1.3 什么是登录帐户？

登录帐户是指能够登录到 Adaptive Server数据库服务器的用户。登录帐户是成为数据库用户的前提和基础。ASE 安装完成后自动建立一个登录帐户 sa. sa 的初始口令为空. sa 具有 SA 和 SSO 的角色.具有 SSO 的登录帐户可以添加其他登录帐户。

1.4 什么是角色，Sybase 具有哪些常用的系统角色以及这些系统角色的作用？

角色是授予指定用户的一组权限.角色允许登录的用户执行必要的管理及安全任务。

Sybase具有的系统角色包括:

System Administrator (SA)
System Security Officer (SSO)
Server Operator (OPER)

SA的权限:

1.服务器的权限
2.磁盘资源分配的权限
3.存取的权限

4.备份和恢复
5.系统管理权限

SSO 的权限:

- 建立服务器的登录帐户并给予初始口令
- 更改登录帐户
- 修改口令
- 设置口令期限
- 建立并管理用户自定义角色
- 授权使用代理授权
- 对其他登录帐户授予及收回 SSO 及 OPER 角色
- 管理审计系统
- 对登录帐户加锁及解锁
- 删除登录帐户

OPER **的权限：**

- 备份及恢复的权限
- 备份及装载所有的数据库
- 备份及装载所有的事务日志

1.5 什么是超级用户，它具有哪些特点？

sa 登录帐户为 Sybase 数据库服务器的超级用户。
sa 超级用户主要具有以下特点：

- 在安装 Sybase 数据库服务器的时候，由系统自动创建。
- 具有 sa_role、sso_role 和 oper_role。
- 不可以被用户删除。
- 其口令初始为空。

1.6 什么是数据库设备？

数据库设备是存储组成数据库的对象的物理资源。“ 设备 “ 不一定指特定的物理设备。它可以是一块盘，如 disk partition**，也可以是操作系统的文件。**

1.7 什么是数据库？

数据库用于存储一套互相关联的对象(如表)的有关信息(数据). 当用户建立自己的数据库时,需要定义如何组织自己的数据.

Adaptive Server **数据库对象包括: 表，规则，缺省，存储过程，触发器，视图等。**

1.8 Sybase 具有那些重要和必要的系统数据库，它们的作用分别是什么？

master ：包含一些系统表，存储管理Adaptive Server所用的数据。
model ：用于创建新数据库的模板数据库。
sybsystemprocs ：包含存储系统存储过程的表。
sybsystemdb ：包含用于分布式事务管理特性的数据。
tempdb ：包含临时表

1.9 什么是数据库用户，以及登录帐户与数据库用户的区别？

一个登录帐户要存取一个数据库，他必须是那个数据库的用户。每个数据库的用户列在 *sysusers* 系统表中。要登录服务器，用户必须是登录帐户。要进入数据库，必须是数据库的有效用户。

1.10 什么是用户定义组？

一个组是包含多个数据库用户的一个集合。当一个组被建立时，它只存在于创建时所在的数据库。组用于对组的所有成员进行授予及回收权限。

每个用户自动属于 *public* 组。除了 *public*组，一个用户只能属于DBO建立的一个组。

1.11 什么是数据库的属主？

数据库的属主是数据库的拥有者。建立数据库的登录帐户是数据库的最初属主。数据库属主可以把属主地位转给其他用户。

每个数据库有一个属主。数据库属主在数据库中可以：

- 设置数据库选项
- 增加及删除数据库用户
- 授予及回收建立数据库对象的权限
- 执行 checkpoint
- 检查数据库一致性
- 执行系统操作员的任务

DBO 可以对数据库的对象行使全部权限，如果：

- DBO 有 sa_role角色, 或
- 对象的属主授予全权给 DBO, 或
- DBO 使用 setuser 转为对象属主的身份

1.12 什么是数据库选项，Sybase 具有那些常用的数据库选项？

数据库选项控制数据库行为的不同方面。

例如：

- **事务的行为**
- **表中列的缺省**
- **用户存取的限制**
- **恢复及 bcp 操作的性能**
- **日志的行为**

数据库选项与配置参数和set设置的选项类似，只是范围不同：

- **配置参数影响服务器范围的行为**
- **数据库选项影响数据库的行为**
- **set 选项影响当前对话或存储过程**

数据库有如下选项：

1. allow nulls by default
2. auto identity
3. dbo use only
4. ddl in tran
5. identity in nonunique index
6. read only
7. single user
8. unique auto_identity index
9. abort tran on log full
10. no chkpt on recovery
11. no free space acctg
12. select into/bulkcopy/pllsort
13. trunc log on chkpt

关于数据库选项的详细使用方法请参考本手册 3.4 数据库选项(dboptio n)的使用与管理。

1.13 什么是数据库一致性检查，Sybase 具有哪些常用的检查命令？

数据库一致性检查通过 dbcc **命令检查数据库的逻辑的和物理的一致性**. dbcc **的两个基本点主要功能是**:

使用 checkstorage **或** checktable **及** checkdb **在页一级和行一级检查页链和数据指针**.

使用 checkstorage, checkalloc, checkverify, tablealloc, **和** indexalloc **检查页分配。**

1.14 什么是 APL 表和 DOL 表，它们的特点和区别？

APL **表是指在事务活动期间,数据页和所有相关的索引叶级页加锁**, **以阻止其它用户的存取**.

Datapages 表是指在事务活动期间,只对数据页加锁, 所有相关的索引叶级页不加锁.当索引叶级页需要修改时,只加Latches 锁防止存取,但这个锁不会保持到事务结束。

Datarows 表是指在事务活动期间, 只对数据行加锁, 其他用户可以使用数据页的其它部分.当索引叶级页需要修改时,只加Latches 锁防止存取,但这个锁不会保持到事务结束。

Datapages 和 Datarows 表就是 data-only locking (DOL) 表. 名称 'data-only locked' 就是指没有索引页上的锁竞争。

1.15 什么是 Sybase 的锁，它具有哪些类型以及这些锁的作用？

Adaptive Server 通过锁的方式保护被活动的事务正在使用的表,数据页, 或数据行. 锁是一种并发控制机制: 它确保事务内及事务间的数据一致性. 在多用户环境中需要锁, 因为在同一时刻多个用户可能对同一数据进行操作。

锁的类型及作用如下:

Shared 锁:

Adaptive Server 对读操作加 shared 锁. 如果一个 shared 锁已经加在一个表、数据页,或数据行,或索引页上,即使加锁的事务仍然是活动的,其它事务也可以获得 shared 锁. 然而,在表或页面或行上的所有 shared 锁被释放之前, 没有事务可以获得对这个表或页面或行的 exclusive 锁. 这意味着多个事务可以同时读表,页面或行, 但没有事务可以对已被加 shared 锁的表,页面或行中的数据进行修改. 需要 exclusive 锁的事务必须等待 shared 锁释放后才能继续.

Exclusive 锁:

Adaptive Server 对数据更改操作加 exclusive 锁. 如果一个事务获得了一个 exclusive 锁, 那么在这个事务结束时释放 exclusive 锁之前,其它事务在相应的表或页面或行上不能获得任何类型的锁. 其它事务必须等待 exclusive 锁释放后才能继续.

Update 锁:

Adaptive Server 在 update, delete, 或 fetch 的初始阶段(当页面或行被读时)加 update 锁. 加 update 锁的页面或行允许加 shared 锁,但是不允许加 update 或 exclusive 锁. Update 锁对避免死锁和锁竞争有帮助. 如果页面或行的内容需要修改, 只要没有其它 shared 锁在上面, update 锁立即升级为 exclusive 锁.

Intent 锁:

Intent 锁指出在一个表上有页级或行级锁. Adaptive Server 对每一个有 shared 或 exclusive 的页或行锁的表加 intent 表锁, 所以 intent 锁可以是 exclusive 锁, 也可以是 shared 锁. 设置 intent 锁可以防止其它后来的事务在有锁住的页的表上获得有冲突的表级锁. Intent 锁持续的时间和事务中页或行锁的时间一样长.

1.16 什么是数据库备份和恢复？

数据库备份是指通过dump命令把数据库中的内容拷贝到磁带或操作系统文件上的过程。

数据库恢复是指通过load命令把数据库的备份内容装载回数据库的过程。

遇到介质故障(如磁盘崩溃)时,仅当用户有数据库的及事务日志的定期备份的情况下才可恢复该数据库.

要完全恢复,则必须定期使用 dump database 和 dump transaction命令备份数据库和日志,发生故障时,才可用 load database 和 load transaction 命令恢复数据库。

2. 日常维护篇

2.1 常规数据库服务器维护的基本要点和步骤？

常规数据库服务器的维护主要是由系统管理员来完成的，主要维护工作包括：

1．对各 ASE 用户担当的角色和特权进行分派

为了保证数据库服务器访问的安全性，以及区分各种登录帐户的工作范围，首先需要系统管理员（这里一般情况是指具有 sa_role 的登录帐户）添加具有一定角色的登录帐户以便进行相关的数据库服务器管理工作。例如对登录帐户的管理；对数据库服务器安全设置的管理；对相关系统参数的管理；以及对于数据库服务器中特殊的 option 的管理，如 DTM,HA 等等。

同时，系统管理员也可以完成对数据库用户的管理工作。例如授予数据库用户查询，插入，删除数据表中数据，以及执行某个存储过程的权限等等。这个工作也可以由数据库属主来完成。

2．管理和监视磁盘空间的使用状况

这里主要是指对数据库空间的管理，包括数据段和日志段；以及数据库设备空间的管理。

系统管理员要定期的对数据库空间和数据库设备空间进行监视及管理。

(1) 数据库空间

可以在 isql环境下执行”sp_helpdb”系统存储过程，显示当前数据库空间的使用情况，观察数据段空间是否足够，是否需要执行”alter database”命令进行扩充；观察日志段空间是否足够，是否需要执行”alter database”命令进行扩充，或者执行”dump tran”命令截断日志。

(2) 数据库设备空间

首先建议用户保存每一次的数据库设备创建的脚本文件，以及每一次每一个数据库设备的使用状况，做到对每一个数据库设备的剩余空间心中有数。因为数据库设备的剩余空间目前只能在 Sybase Central的管理工具中有明显标识，而在 isql 环境下没有很直接的命令或者存储过程查看数据库设备的剩余空间。如果没有数据库设备的使用记录，又无法调用 Sybase Central图形化管理工具，则会给系统管理员的管理工作带来不便。

当数据库服务器中几乎没有可以使用的数据库设备时，就需要在 isql环境下执行”disk init”命令创建新的数据库设备以备用。

3．进行日常的数据库备份与恢复

首先需要系统管理员根据系统运行状况以及业务需求指定相应的数据库备份与恢复策略，之后以该策略为指导进行日常数据库备份与恢复。如何指定备份策略以及进行数据库备份与恢复的必要步骤，请参考例行数据库备份恢复策略和必要步骤。

4．诊断系统故障

在系统诊断的过程中，首先需要具有一定系统维护经验的系统管理员确认是操作系统的问题，还是 Sybase数据库服务器的问题。以下简要步骤只是说明了，当碰到数据库服务器的问题时该如何做。

(1)详细记录出现系统故障时的现象，以及在应用过程中可能出现在客户端的错误提示信息

(2)查看 Sybase 错误日志文件：$SYBASE/ASE-12_5/install/server_name.log,查找不正常的信息提示或者出现的错误代码。如果此时能够确认是数据库本身的问题，例如数据页或者索引损坏，那么可以先执行 DBCC 命令进一步确认问题的确切所在。（DBCC 命令的使用，请参照后面的相关章节）

(3)按照错误日志文件中所提示的错误代码，查看《Troubleshooting and Error Messages Guide》并按书中所提示的纠错步骤进行处理

(4)在自行无法解决的情况下，可以联系Sybase 工程师寻求帮助，同时提供：

——详细的产品版本信息
——详细的现象描述
——详细的错误信息提示
——必要的查询输出结果
——完整的数据库服务器错误日志文件

5．调整 ASE 的性能

这是一项需要长时间的，不断尝试的，不断测试的，不断优化的循序渐进的过程。针对不同的数据库服务器的运行环境，不同的业务需求就会出现不同的性能要求。

我们说调优是一门艺术，就是要在不断磨合的过程中以求得数据库服务器的最佳使用状态。

Sybase 会提供系列的性能分析的工具来帮助用户实现最有效的调优方法。具体说明请参照查找定位数据库服务器性能问题的常用工具和方法。

2.2 例行数据库备份恢复策略和必要步骤？

在例行数据库备份之前，首先要制定有效的数据库备份策略，即应该多长时间备份一次数据库。这主要是基于当发生系统故障时(例如磁盘故障) 应用系统允许丢失多长时间的数据。

例如某个系统允许丢失一个小时的数据，那么可以考虑在每天的固定时刻对整个数据库做一次备份，并每一个小时对数据库日志做一次备份。那么当系统故障排除之后，就可以用最近一次的数据库备份以及直到系统故障发生时的日志备份来对数据库数据进行恢复。

这里要注意的是：第一，保留系统出现故障前的最近一次完整的有效的数据库备份，这很重要；第二，在做数据库恢复时，要先恢复完整的数据库备份，再依次恢复每一小时的日志备份，直到出现系统故障前的最后一次日志备份。

在备份数据库时应该注意的问题：

1．在做数据库备份操作之前，应对数据库做 dbcc 检查，以确保备份的数据库是完好的数据库。如果数据库有损坏，备份时可能不会报错，但将来可能无法恢复。

2．通常情况下，只有发生严重故障后，需要恢复数据库备份时才进行恢复数据库的操作。但是 Sybase 建议偶尔对备份的数据库恢复到一个测试环境上，是有利于保证备份工作过程正常以及数据库备份有效的很好的测试方法。

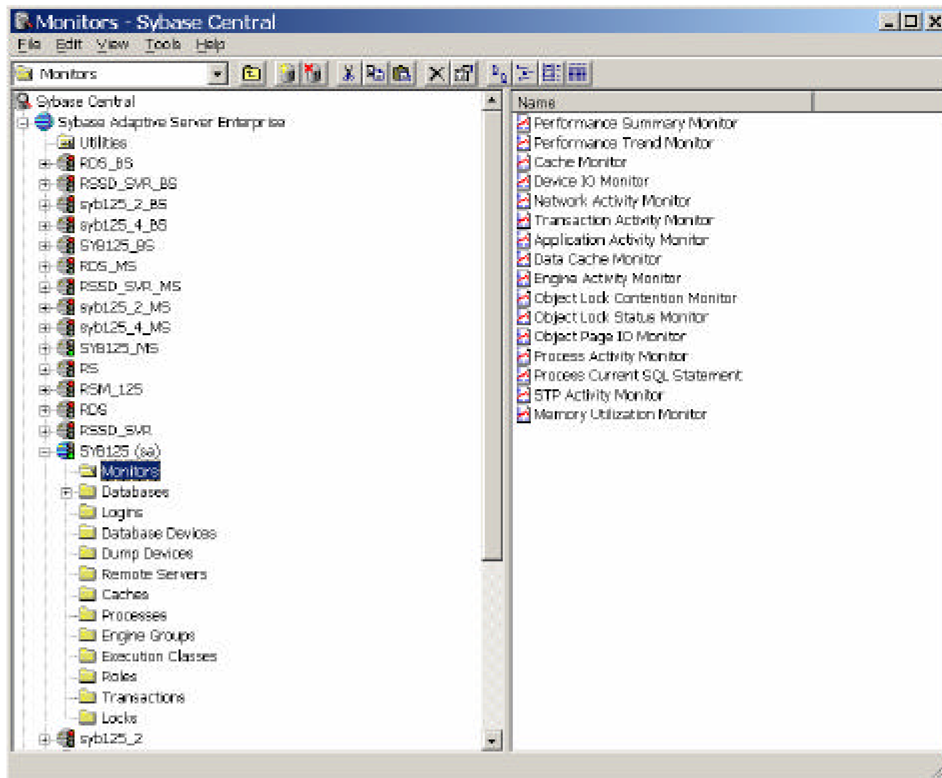
3．Master 数据库的备份也不能忽略。每次master 数据库的内容有变动时，例如增加login，增加设备，增加用户数据库等，应及时备份 master 数据库。

用户可以根据自己的应用制定可行的数据库备份恢复的方案。SYBASE 数据库的备份主要是通过 dump 命令来实现的，分为数据库备份和日志备份；数据库的恢复主要是通过 load 命令来实现的，也分为数据库恢复和日志恢复。语法命令，请参照以后的相关章节。

2.3 查找定位数据库服务器性能问题的常用工具和方法？

1．**实用工具类**

(1) **在** Sybase Central**图形化管理工具中，使用** Monitor Server **监视数据库服务器性能**



从上图中看到，在 Sybase Central**图形化管理工具右边所列出的条目就是所有可以以图形化的方式监测的数据库服务器的性能指标。**

(2) isql –p

在使用 isql **登录数据库服务器时，加**-p **选项，那么之后执行的每一条** SQL **语句都可以得到它的执行时间，这里所记的是** CPU **的时钟时间。**

(3) reorg

该命令用于优化 DOL **表的表空间及其索引。使用方法，请参考如何执行 reorg 系列命令来优化 DOL 表及其索引**

(4) optdiag

该命令用于显示数据库服务器中的表和索引的统计值信息。使用方法，请参考如何执行 optdiag 命令来监控表和索引的物理使用情况

2．Set **命令**

(1) statistics io

执行方法：

```
1>  set statistics io on
```

```
2> go
```

之后所执行的任何 SQL 语句，执行完毕后，都会得到物理读、写，逻辑读、写的次数

(2) statistics time

执行方法：

```
1> set statistics time on
2> go
```

之后所执行的任何 SQL 语句，执行完毕后，都会得到执行时间，以 CPU 的时钟时间记时。

(3) showplan

执行方法：

```
1> set showplan on
2> go
```

之后所执行的任何 SQL 语句，执行完毕后，显示由调优器所选择的执行计划，包括执行步骤，索引的使用，I/O 的使用状况等等。

(4) showplan and noexec

执行方法：

```
1> set showplan,noexec on
2> go
```

之后所执行的任何 SQL 语句，不需要执行，就直接显示由调优器所选择的执行计划，包括执行步骤，索引的使用，I/O 的使用状况等等。

3．系统存储过程

(1) 进程行为查看类

① sp_who

请参看如何查看当前数据库中的进程信息

② sp_lock

请参看如何查看当前数据库中锁发生的情况

③ sp_object_stats

执行方法：

```
1> sp_object_stats “00:20:00”,20
2> go
```

以上例子将输出在当前执行这条语句的数据库中，在 20 分钟内锁竞争最激烈的前 20 张表

④ sp_showplan

执行方法：

```
1> sp_showplan 20, null, null, null
2> go
```

以上例子输出 20 号进程所执行的 SQL 语句的执行计划。

(2) 空间使用类

① sp_spaceused

执行方法：

```
1> sp_spaceused table_name
2> go
```

输出指定表的空间使用情况。

② sp_helppartition

执行方法：

1> sp_helppartition table_name
2> go

输出指定表的分区使用情况

③ sp_estspace

执行方法：

1> sp_estspace table_name,nr_of_rows
2> go

输出指定表(table_name)**中如果存在指定行数**(nr_of_rows)**的数据，估计的空间使用情况。**

④ sp_helpsegment

执行方法：

1> sp_helpsegment seg_name
2> go

输出指定段的使用情况。

⑤ sp_helpcache

执行方法：

1> sp_helpcache cache_name
2> go

输出指定缓存的使用情况。

(3) **系统配置类**

① sp_configure

请参看如何执行 sp_configure 系统存储过程来查看系统参数

② sp_cacheconfig

请参看如何配置用户自定义命名缓存以及缓冲池

③ sp_poolconfig

请参看如何配置用户自定义命名缓存以及缓冲池

(4) **任务行为查看类**

① sp_sysmon

请参看如何使用 sp_sysmon 存储过程来查看当前数据库的性能情况

② sp_monitor

执行方法：

1> sp_monitor
2> go

输出 CPU,I/O **使用情况**

4．dbcc trace**命令**

(1) dbcc trace(3604，302，310)

执行方法：

1> dbcc trace(3604，302，310)
2> go

输出更为详细的 SQL **语句执行计划**

2.4 开发数据库应用时需要开发人员重点关注的几个问题？

通常，开发一个应用包括如下几个阶段或步骤：

1．**弄清业务需求，定义业务规则**
2．**规划应用开发环境，特别是定义软件规范说明**
3．**完成数据库的逻辑和物理设计**
4．**应用程序设计（编程和调试）**
5．**运行与维护**

在程序设计时，要注意如下两个问题：

1．**对于复杂的业务规则应放在服务器上去实现，而不是在客户机上实现，这样做将使程序设计更省力，且效果更好。**
2．**由客户机程序将“批”发送到服务器，要求在程序结构中包含对错误的处理和对事务的管理能力。**

2.5 如何在 Windows 平台上启动和关闭 Sybase 数据库服务器？

有两种常用方式：

1．**使用** Sybase**工具：**Sybase Central
打开 Sybase Central-->**找到所需要启动的** Sybase **数据库服务器的名称**-->**鼠标右键单击该名称**,**出现弹出框**
启动：–>**单击**”start”
关闭：–>**单击**”stop”
2．**使用** Windows **操作系统提供的服务管理**
打开“Control Panel”**中的**“Service”--> **找到所需要启动的** Sybase **数据库服务器的服务名称**–>**鼠标右键单击该名称**,**出现弹出框**
启动：–>**单击**”start”
关闭：–>**单击**”stop”

2.6 如何在 UNIX 平台上启动和关闭 Sybase 数据库服务器？

启动：
使用 SYBASE **用户在操作系统提示符下执行：**
startserver –f RUN_servername
关闭：
isql **登录到数据库服务器，执行：**

1> shutdown
2> go

注：如果需要关闭备份服务器，执行：

1> shutdown SYB_BACKUP
2> go

2.7 如何使用交互式管理工具 isql?

在 UNIX **操作系统平台，使用** SYBASE **用户执行该命令；或者在** Windows **平台的** DOS **窗口中执行该命令。具体命令的通常使用方法如下所示：**

isql –Uuser_name –Ppassword –Sserver_name
其中 user_name**登录帐户名，例如** sa
password:**登录帐户的口令，例如** sa **的口令** 123456

server_name 数据库服务器的名称或者是在 Dsedit 使用程序中定义的相应的服务器名称，例如 SYB125

正确使用这个命令后，会进入 1>提示符，而没有任何错误、信息提示。

2.8 如何在 Windows 操作系统上安装 Open Client12.0

1. 选择 Open Client 的正版。
2. 安装 Open Client 时，如果安装程序提示是否覆盖相关.DLL 文件时，一定选择 NO。
3. 编辑客户端 win98/win95 的 autoexec.bat 和 configure.sys 文件（在其中添加自动执行 SYBASE 设置环境变量的批处理文件 sybase.bat，以及添加 SYBASE 的路径，DLL 路径，BIN 路径）。

#假设 SYBASE 安装在 C:\SYBASE

(1) 编辑 autoexec.bat：

```
path='c:\sybase;c:\sybase\ocs-12_0\dll;c:\sybase\ocs-12_0\bin';
call c:\sybase\sybase.bat
```

(2) 编辑 config.sys：

```
shell=c:\windows\command.com /p /e:2048
files=50
```

2.9 如何使用图形化管理工具 Sybase Central？

这里主要介绍 windows 平台的 java 版的 **Sybase Central** 和 windows 版的 **Sybase Central**。

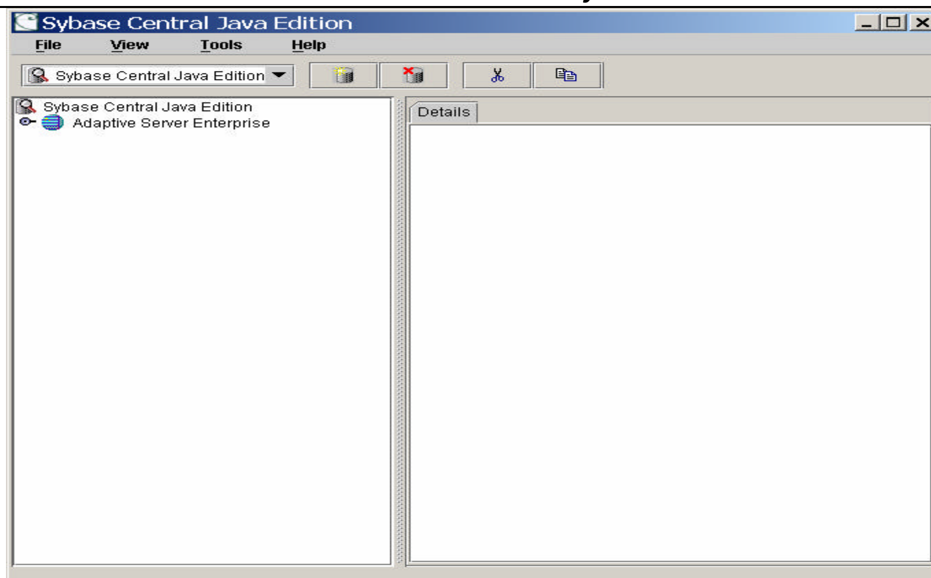
Sybase Central 是一个用于管理数据库及相关产品的简单易用的图形化管理工具。它能够帮助您管理数据库服务器，数据库对象以及执行一些系统管理的任务，同时为每一个任务都能够提供便于参考的循序渐进的图形化的步骤说明，使得系统管理员在管理过程中游刃有余。

下面介绍使用方法。

java 版的 **Sybase Central-------**

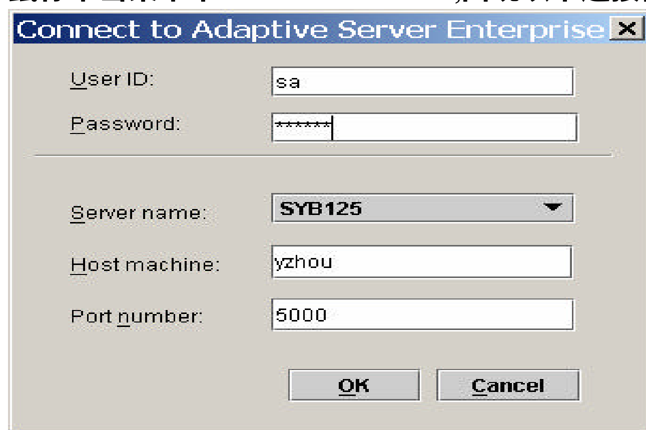
1.打开 **Sybase Central** 图形化界面：

在“开始”菜单中寻找：开始-->程序-->Sybase-->**Sybase Central Java** 版，鼠标单击 **Sybase Central Java** 版，出现以下界面：

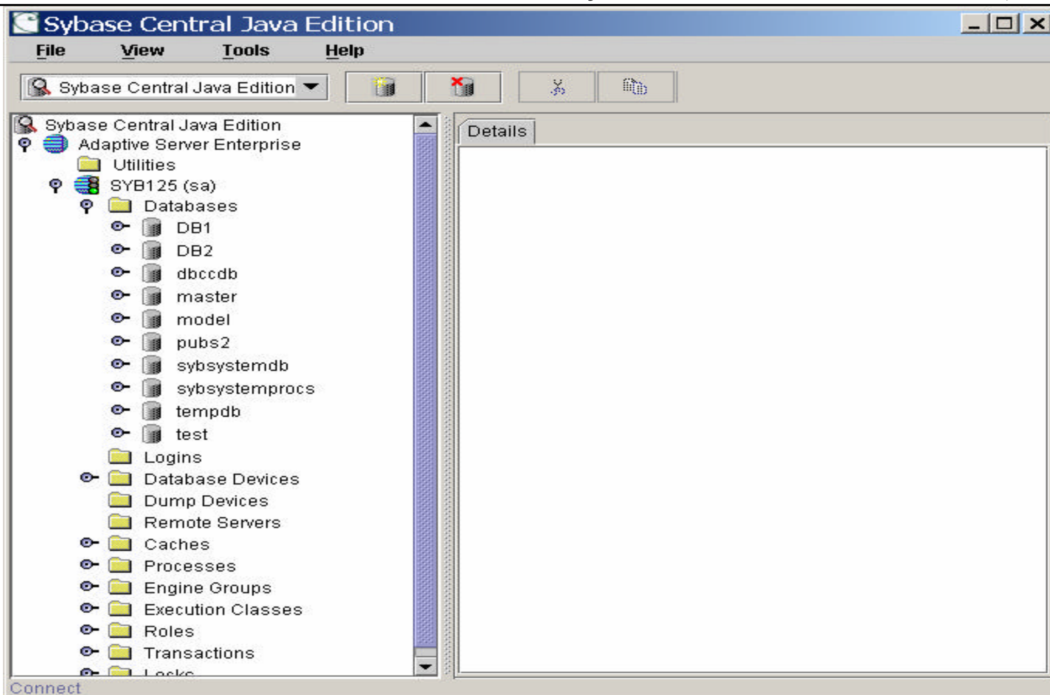


2。**连接数据库服务器**

鼠标单击菜单中：Tools-->Connect,**出现以下连接窗口：**



输入登录帐户名，口令，选择需要连接的数据库服务器名称，之后鼠标单击“OK”**按钮，即可看到如下界面：**



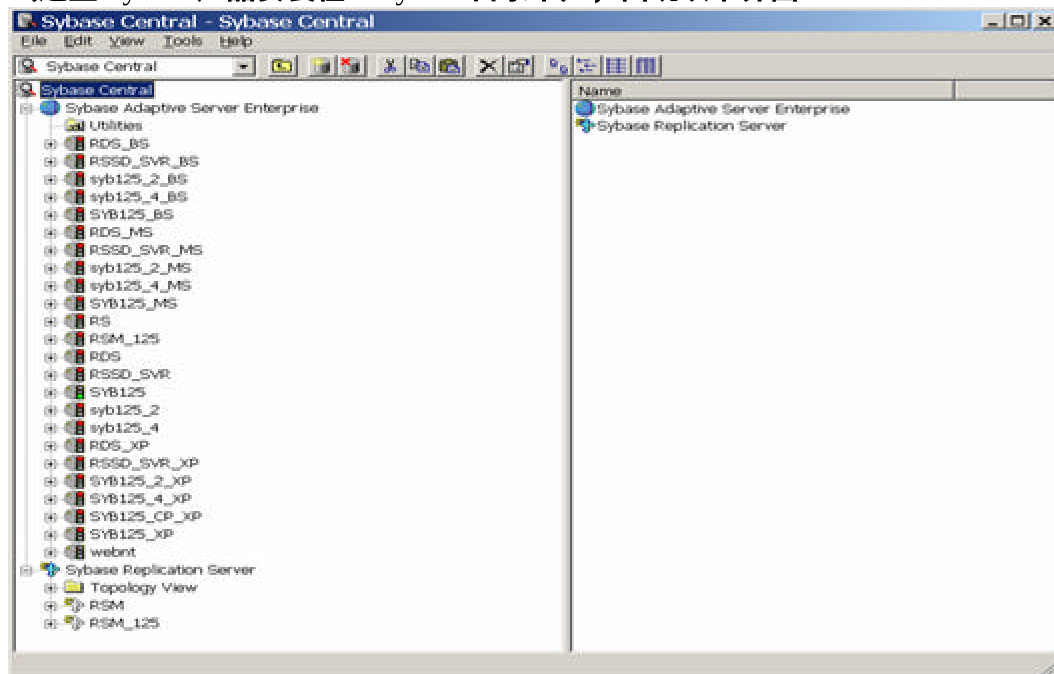
现在您就可以对您的数据库服务器进行需要的管理工作了。

windows **版的 Sybase Central----**

1.**打开 Sybase Central图形化界面：**

在 Sybase**产品的安装目录下执行**"E:\sybase\Sybase Central 3.2\win32\scview.exe"

（这里 Sybase **产品安装在** e:\sybase **目录下），出现以下界面：**

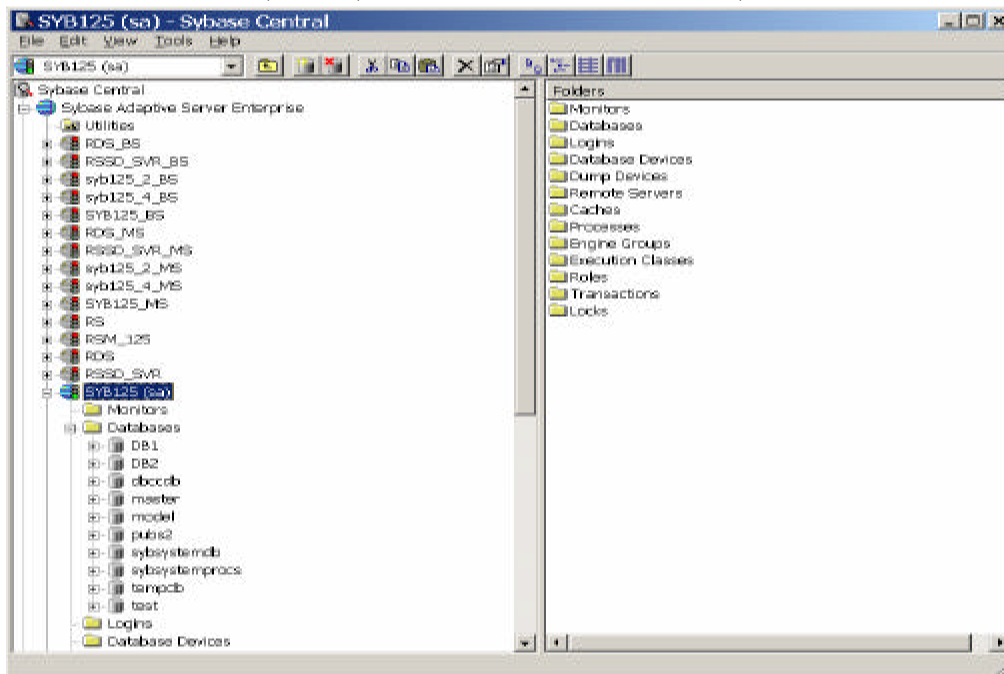


2．**连接数据库服务器**

鼠标双击需要连接的数据库服务器名，出现登录界面：



输入登录帐户名，口令，之后鼠标单击“OK”按钮，即可看到如下界面：



现在您就可以对您的数据库服务器进行需要的管理工作了。

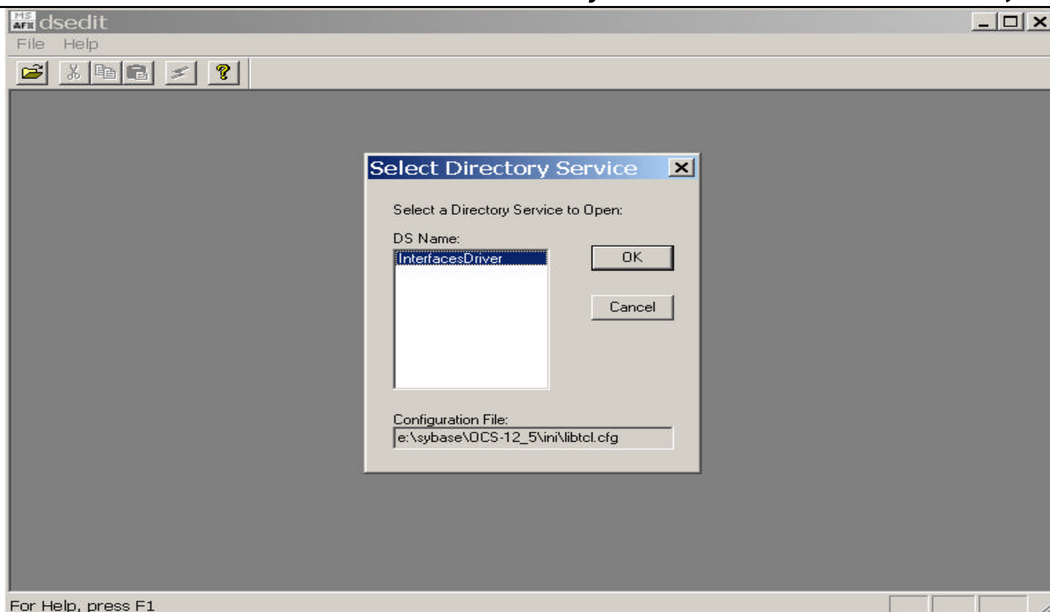
2.10 如何使用图形化管理工具 Dsedit 实用程序？

Dsedit 实用程序是客户端用于配置服务器端数据库服务器连接属性的配置工具。在客户端访问服务器端数据库服务器之前，数据库服务器连接属性的配置是必须的也是必要的，那么连接属性的配置将使用 Dsedit 实用程序来完成。

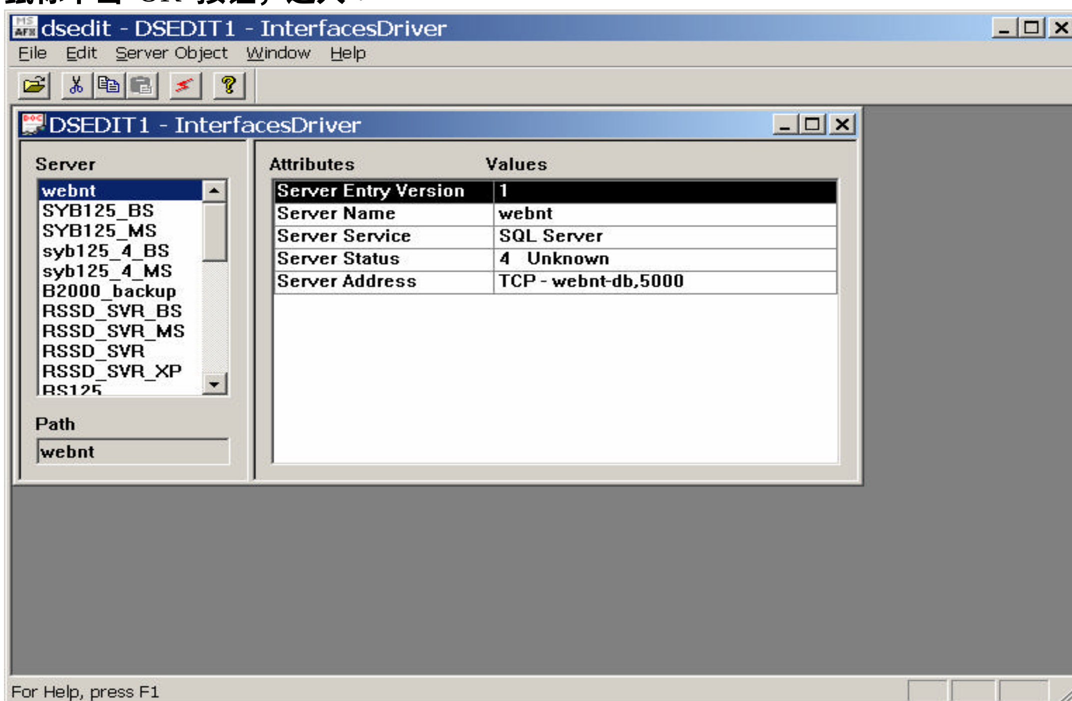
这里主要介绍 windows 平台的 Dsedit 实用程序的使用方法。

1．打开 Dsedit 实用程序图形化管理工具：

在“开始”菜单中寻找：开始-->程序-->Sybase-->Dsedit 实用程序，鼠标单击 Dsedit 实用程序，出现以下界面：

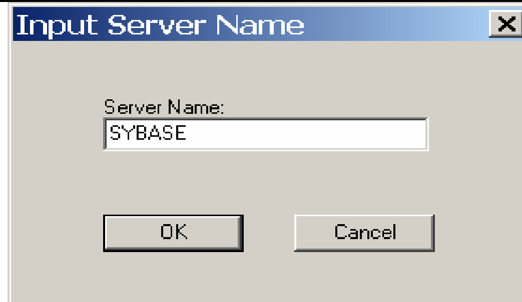


鼠标单击"OK"按钮，进入：

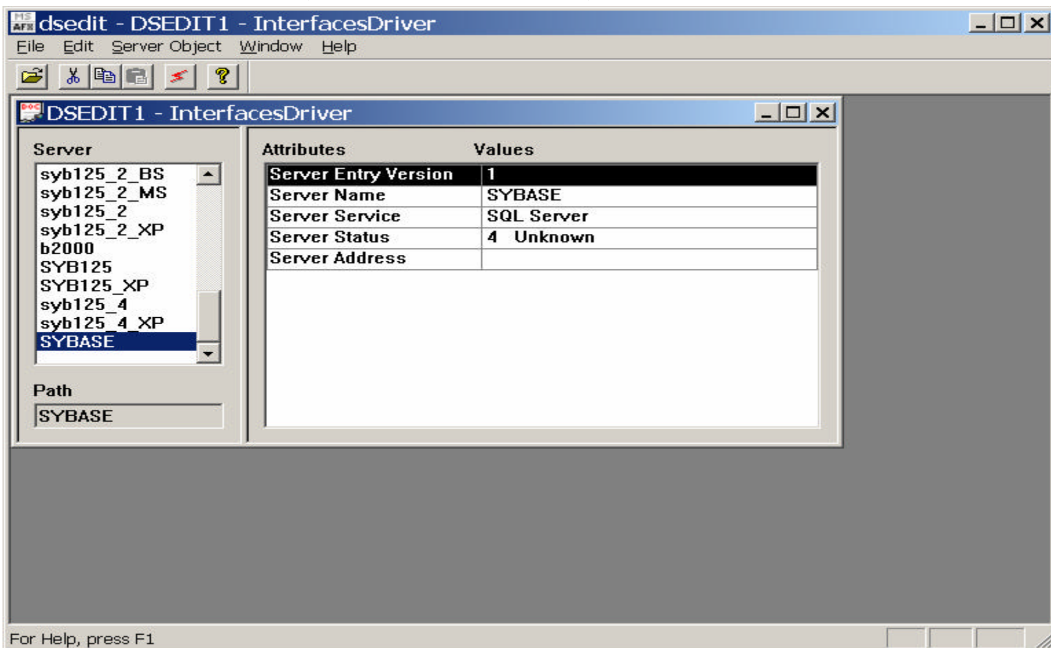


2. **添加数据库服务器名称：**

鼠标单击菜单中：Server Object-->Add,**出现以下窗口：**

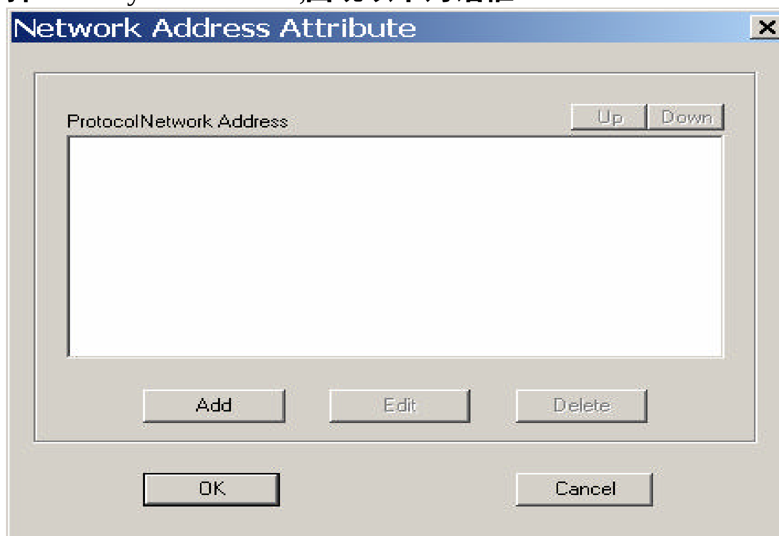


输入数据库服务器名称，鼠标单击”OK”按钮，出现以下界面：

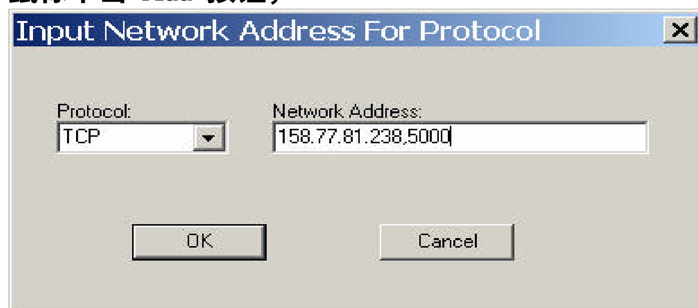


3. **添加数据库服务器地址：**

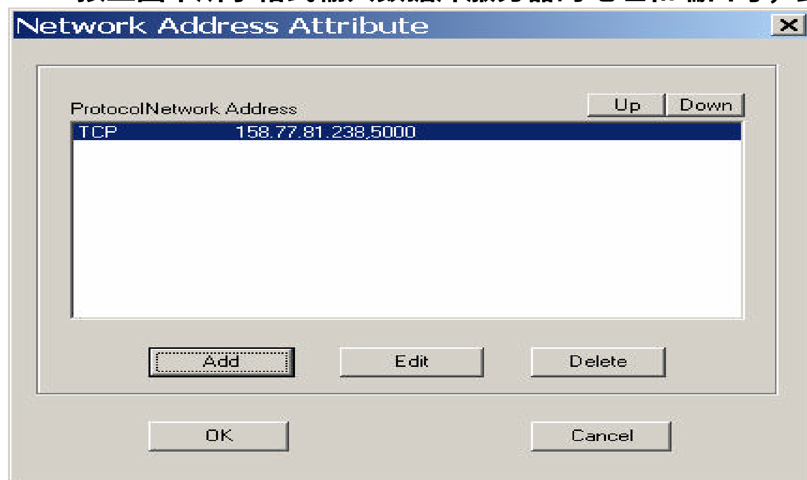
用 鼠标选择 Server Address **条目，再鼠标右键单击该条目，出现弹出框，选择**”Modify Attribute…”,**出现以下对话框：**



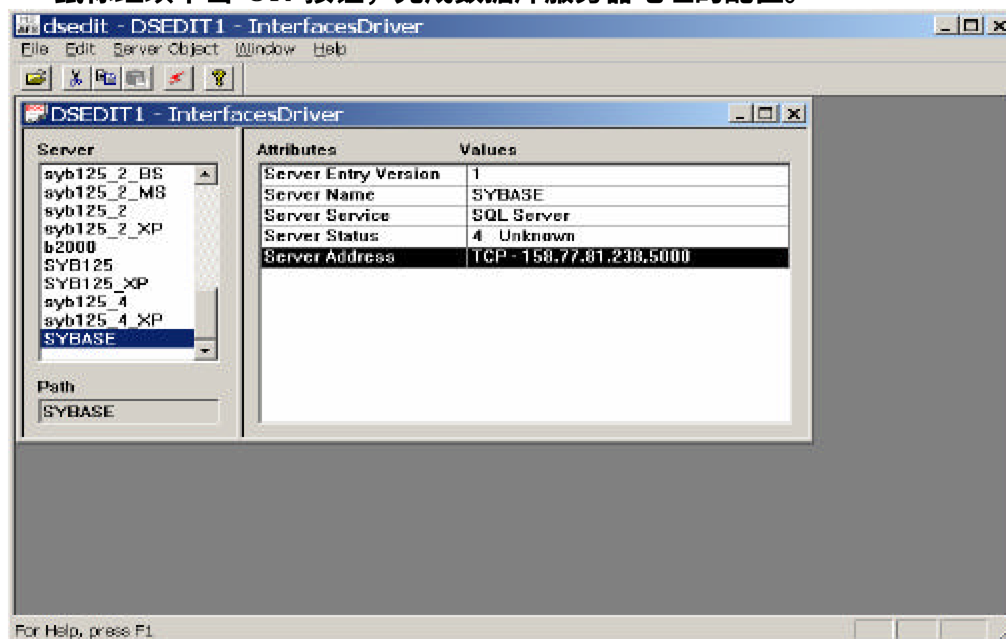
鼠标单击"Add"按钮，



按上图中所示格式输入数据库服务器的地址和端口号，鼠标单击"OK"按钮，



鼠标继续单击"OK"按钮，完成数据库服务器地址的配置。



关闭该图形化窗口，即完成数据库服务器连接属性的配置。

2.11 如何创建登录帐户？

在 isql 环境下执行：

1> **sp_addlogin** login_name,password
2> go

例：

1> sp_addlogin user1,” 123456”
2> go

(注：该存储过程还提供其它参数，如需使用，请参阅资料《Reference Manual》)

2.12 如何修改自己以及其登录帐户的口令？

在 isql环境下执行：

1> **sp_password** caller_password,new_password,login_name
2> go

例：

1> sp_password “111111” ,” 222222” ,user1
2> go

其中：

caller_password：登录时所使用的 login 的口令。在以上例子中，如果登录时使用的是 sa，那么“ 111111 ”就是 sa 的口令；如果登录时使用的是 user1，那么“ 111111 ”就是 user1的登录口令。

new_password：需要更新的 login 的新口令。在以上例子中，“ 222222 ”是 user1的新口令。

login_name：需要更新的 login 名称。如果该参数不提供，则默认更改登录时所使用的 login 名称。

2.13 如何查看登录帐户的信息？

在 isql环境下执行：

1> **sp_displaylogin** login_name
2> go

例：

1> sp_displaylogin user1
2> go

其中：

login_name:为需要查看的 login 名称。如不指定，则默认为登录时所使用的 login 名称。

2.14 如何设置登录帐户的缺省数据库？

方法一：

在创建登录帐户时即可指定，具体方法如下，在 isql环境下执行：

1> **sp_addlogin** login_name,password,default_db
2> go

例;

1> sp_addlogin user1,”123456”,pubs2
2> go

(注：这里如果不指定，缺省数据库为 master)

方法二：

为已经创建好的登录帐户修改缺省数据库，具体方法如下，在 isql**环境下执行：**

1> **sp_modifylogin** login_name,**" defdb"**,db_name
2> go

例：

1> sp_modifylogin user1,"defdb",pubs2
2> go

上例中，将 login **是** user1**的缺省数据库设置为** pubs2**数据库。**

2.15 如何为登录帐户授予系统角色？

方法一：

在 isql **环境下执行：**

1> **sp_role "grant"**,role_name,login_name
2> go

例：

1> sp_role "grant" ," sybase_ts_role",user1
2> go

上例中，将系统角色 sybase_ts_role **授予登录帐户** user1。

方法二：

在 isql **环境中执行：**

1> **grant role** role_name **to** login_name
2> go

例;

1> grant role "sybase_ts_role " to user1
2> go

2.16 如何创建数据库设备？

在 isql **环境中执行：**

1> disk init
2> **name**="device_name",
3> **physname**="physicalname",
4> **vdevno**=virtual_device_number,
5> **size**=device_size

例：

1> disk init
2> name="DATA1",
3> physname="c:\sybase\data\data1.dat" ,
4> vdevno=6,
5> size=5120

以上例子中，创建了一个名为 DATA1**的设备，设备文件为** c:\sybase\data\data1.dat,**虚拟设备号为** 6**，大小为** 10M(size **默认大小的单位为** 2K)

注：在 ASE125**以上版本中，**device_size **单位可以自定义为** K,M,G。**使用这些单位时，需要用双引号将** device_size **的值引起来，例如：**"100K" ," 10M" ," 1G"。

2.17 如何把数据库设备设置为缺省设备？

在 isql **环境下执行：**

1> **sp_diskdefault** device_name,**defaulton**
2> go

例：

1> sp_diskdefault "DATA1",defaulton
2> go

以上例子中，将数据库设备"DATA1" **设置为缺省设备。**

2.18 如何镜像数据库设备和取消数据库设备镜像？

镜像数据库设备：

在 isql **环境下执行：**

1> disk mirror
2> **name** ="device_name" ,
3> **mirror**=" pathname"
4> go

例：

1> disk mirror
2> name="DATA1",
3> mirror="c :\sybase\data\DATA1_M.dat"
4> go

以上例子中为数据库设备 DATA1**创建镜像** DATA1_M.dat.

取消数据库设备镜像：

1> disk unmirror
2> **name** ="device_name" ,
3> **side** =" primary"|" secondary",
4> **mode**=retain|remove
5> go

例：

1> disk unmirror
2> name="DATA1",
3> side=" primary",
4> mode=remove

其中：

side**：指定所要取消镜像的是** primary**（主设备），还是** secondary**（镜像设备）。如果不指定，默认取消** secondary**（镜像设备）。**

mode**：指定取消镜像这个操作是** remove**（永久操作），还是** retain **（临时操作）。如果不指定，默认为** retain **（临时操作）。临时取消镜像的操作，可以使用**"**disk remirror**"**命令恢复其镜像。**

以上例子中，永久取消数据库设备 DATA1**的主设备镜像。**

附：

使用 disk remirror **命令恢复被临时取消的数据库设备镜像。**

1> disk remirror
2> **name** ="device_name"
3> go

2.19 如何创建数据库？

在 isql **环境下执行：**

1> create database database_name
2> on data_device_name=data_size
3> log on log_device_name=log_size
4> go

例：

1> create database DB1
2> on DATA1=100
3> log on LOG1=20
4> go

以上例子中创建了一个数据为 100M**，日志为** 20M**，总大小为** 120M **的数据库** DB1。

注：在 ASE125**以上版本中，**data _size **和** log_ size **单位可以自定义为** K,M,G。**使用这些单位时，需要用双引号将** data_size,log_size **的值引起来，例如：**"100K" ," 10M" ," 1G"。

2.20 如何在 Sybase Central 中配置 Monitor Server?

1.运行$SYBASE\ASEP_Win32\jdk1_1_8-win.exe,安装路径 c:\JDK1.1.8

2.配置环境变量
(1)JAVA_HOME=c:\JDK1.1.8
(2)CLASSPATH=d:\sybase\ASEP\3pclass.zip;d:\sybase\ASEP\monclass.zip;c:\JDK1.1.8\lib;c:\JDK1.1.8\lib\classes.zip
(3)在 PATH 环境变量中加入：c:\JDK1.1.8\bin;c:\sybase\Sybase Central 3.2\win32 且将 c:\JDK1.1.8\bin 放在开始处，并且确认 PATH 环境变量中包含 c:\WINNT 或者 c:\Windows.

3.复制文件
(1) 将 c:\JDK1.1.8\bin 下地文件 jre.exe 和 jrew.exe 复制到 c:\WINNT 或者 c:\Windows(视不同操作系统决定)目录下
(2) 复制$SYBASE\ASE_12-5\bin\libunic.dll 到 c:\WINNT\system32 目录下

4.创建 Sybase Central 快捷方式
在桌面上创建 Sybase Central 快捷方式之后，修改该快捷方式的属性：
在属性对话框中选择"shortcut"页，在"start in:"的输入框中填入：
c:\jdk1.1.8\bin

5．重启机器

2.21 如何配置用户自定义命名缓存以及缓冲池？

定义命名缓存：
在 isql**环境中执行：**

1> **sp_cacheconfig** cache_name,size

2> go

例：

1> sp_cacheconfig cache_a,"100K"
2> go

其中：

size**：命名缓存的大小，单位可以是** P/K/M/G**，默认单位为** K。

上例中创建了一个 100K **的命名缓存** cache_a。

(**注：该命令还提供其它参数，如定义命名缓存的类型等。若需使用，请参阅资料**《Reference Manual》)

定义缓冲池：

在 isql**环境中执行：**

1> sp_poolconfig cache_name,mem_size,config_pool
2> go

例：

1> sp_poolconfig cache_a,"100M" ," 16K"
2> go

其中：

mem_size**：缓冲池的大小，单位可以是** P/K/M/G**，默认单位为** K**，最小指定值为** 512K。

config_pool**：**I/O **大小，可以为** 2K,4K,8K,16K

上例中创建一个大小为 100M **的缓冲池，其** I/O **大小为** 16K。

(**注：该命令的其它用法，请参阅资料**《Reference Manua》)

2.22 如何扩展数据库？

扩展数据空间：

alter database *db_name* on *device_name*

扩展日志空间：

alter database *db_name* log on *device_name*

2.23 如何设置数据库为单用户模式？

use master
go
sp_dboption *db_name*, "**single user**" , true
go
use db_name
go
checkpoint
go

2.24 如何把数据库设置为自动截断事务日志？

use master
go
sp_dboption *db_name*, "**trunc log on chkpt**", true
go
use *db_name*

go
checkpoint
go

2.25 为了可以在用户事务中执行数据定义语言，应如何设置数据库选项？

use master
go
sp_dboption *db_name*, "**ddl in tran** ", true
go
use *db_name*
go
checkpoint
go

2.26 如何设置数据库选项可以在用户表中自动添加序号生成器？

use master
go
sp_dboption *db_name*, "**auto identity**" , true
go
use *db_name*
go
checkpoint
go

2.27 如何为数据库创建数据库用户？

语法：sp_adduser loginame [, name_in_db [, grpname]]

举例：

1.sp_adduser margaret

将"margaret"**增加为数据库用户。他的数据库用户名与** Adaptive Server **的登录名（**login name**）相同**, **并且属于缺省组**"public"**。**

2.sp_adduser haroldq, harold, fort_mudge

将"haroldq"**增加为数据库用户。当** "haroldq"**使用当前数据库时，他的名字为** "harold"**。该用户属于"** fort_mudge **"组，同时也属于缺省组** "public"**。**

2.28 如何创建用户自定义组？

语法：sp_addgroup grpname

举例：

1.sp_addgroup accounting

增加一个名为" accounting **"的组。**

2.29 如何改变用户自定义组中数据库用户？

语法：sp_changegroup grpname, username

举例：

1.sp_changegroup fort_mudge, albert

用户 "albert"**现在成为**"fort_mudge"**组的成员。**

2.30 如何查看数据库空间的大小？

举例：

1.**查看全部数据库的大小：**

```
sp_helpdb
name          db_size   owner dbid created       status
------------- --------  ----- ---- ------------- -----------------
master        5.0 MB    sa       1 Jan 01, 1900  no options set
model         2.0 MB    sa       3 Jan 01, 1900  no options set
pubs2         2.0 MB    sa       6 Sep 20, 1995  no options set
sybsystemprocs 16.0 MB  sa       4 Sep 20, 1995  trunc log on chkp
tempdb        2.0 MB    sa       2 Sep 20, 1995  select into/bulkcopy
```

Displays information about all the databases in Adaptive Server.

2.**查看指定数据库的大小**

```
sp_helpdb pubs2

name  db_size  owner  dbid  created         status
----- -------  -----  ----  ------------  ----------------------
pubs2  2.0 MB   sa      4   Mar 05, 1993   abort tran when log full
device_fragments      size      usage           free kbytes
-----------------     ------    ------------    -------------
master                2.0 MB    data and log         576
name    attribute_class attribute          int_value char_value   comments
------- --------------- ------------------------ ----------- --------
pubs2   buffer manager  cache binding          1   pubs2_cache    NULL
```

Displays information about the pubs2 database.

2.31 如何改变数据库的属主？

语法：sp_changedbowner loginame [, true]

举例：

1．sp_changedbowner albert

用户 "albert"**成为当前数据库的属主。**

2.32 如何执行 sp_configure 系统存储过程来查看系统参数？

1．**查看全部参数的配置：**

```
isql -Usa –P –Ssevername
1> sp_configure
2> go
```

2．**查看指定参数(如：查看"用户连接数")**

```
isql -Usa –P –Ssevername
1> sp_configure “number of user connections”
```

2>go

2.33 如何为当前的 Sybase 数据库服务器设置最大可用内存？

在 ASE12.5 中，当前 SYBASE 数据库服务器可用的最大内存由参数“max memory”配置，其含义是 ASE SERVER 可分配的最大共享内存（pre-ASE12.5,使用参数 total memory）。ASE12.5 采用动态分配内存机制，在安装时由于没有什么进程、任务需要用到内存，此时系统分配的共享内存是很小（可用 sp_configure “total physical memory”查看当前的内存使用情况）。因此即使我们将“max memory”配置到很大的值，比如 2G、4G，但 ASE SERVER 在实际需要时并不一定能分到“max memory”指定的内存。因此，在安装时，必须先将“allocate max shared memory”设置为 1（在 SERVER 启动时就划分”max memory”的值的内存），再启动 SERVER,如果启动正常，那么说明“max memory”的值是系统可以接受，此时可以把“allocate max shared memory”设置为 0，让 ASE SERVER 动态分配内存；如果启动失败，则要查看操作系统有关内存的配置参数是否正确设置。

2.34 影响数据库服务器内存配置的基本参数

设置 Adptive Server12.5 的内存配置时，使用 sp_configure 命令，用具体值指定每一项内存需求，例如过程缓存和缺省数据缓存的大小。以下 3 个参数影响内存的分配（不同于以前版本）：

max memory 该参数是以 2K 为单位的数值，决定可以分配给 Adaptive Server 的最大内存数。为动态参数，改变后不必重起 Adaptive server。

allocate max shared memory 该参数有 0 和 1 两个值。决定 sybase server 启动时，是否一次分配给 Adaptive server 的内存为 max memory 指定大小。值为 1 时启动 sybase server，一次分配 max memory 大小的内存；值为 0 时启动 sybase server，一次只分配按实际参数配置所需要的内存。例如，max memory 为 500MB，若设置该参数为 0（缺省值）， Adaptive Server 的具体配置参数和核心所需内存仅 100M，则 Adaptive server 启动后只从 OS 获取 100MB 内存，仅当动态参数改变需要额外内存时才考虑从 OS 获取所需内存，最大不超过 max memory 大小；若设置该参数为 1，则 Adaptive Server 在启动时一次即获取 500MB 内存，而不管空闲内存是多少。

Dynamic allocation on demand 该参数用以确定有新的内存请求时，是立即获得内存资源还是仅在需要时分配。是动态参数，取值范围为 0 和 1。例如，假设 number of user connections 原值为 50，改为 150，因每个用户连接需要 112K 内存，则（150－50）个用户需要使用的内存数量为 11MB（100 x 112/1024）。若设置该参数值为 1，当有第 51 个连接需求时，才分配一个连接需要的内存，也就是说是随着真正的

需要逐渐获取连接所需内存，直到 150 个连接局限；如果设置该参数值为 0，Adaptive Server 立即为用户连接分配（150一 50）个用户需要的 11MB 内存。

2.35 如何为当前的 Sybase ASE 12.5 设置可用的数据缓存？

total data cach size 以 K 字节显示当前有效的数据缓存大小，为只读参数，只能使用 sp_configure查看，不能配置。配置数据缓存，必须使用 sp_cacheconfig。ASE12.5 的 default data cache 必须手动重配，默认值 8M 不能满足性能需要。注意该参数为静态参数，必须重起 ASE。

示例：为 Adaptive Server 配置 100MB 数据缓存

sp_cacheconfig " default data cache"," 100M"

2.36 如何为当前的 Sybase ASE 12.5 设置可用的过程缓存？

参数 **procedure cach size** （以 2k 为单位）决定了过程缓存的大小，该参数使用 sp_configure配置，为动态参数。

示例：为 Adaptive Server 配置 100MB 过程缓存

sp_configure "procedure cache size"," 100M"

2.37 如何为当前的 Sybase 数据库服务器设置可用的最大数据库设备数量？

Number of device参数控制了 Adaptive Server 能够使用的数据库设备的数目。

示例：最多允许使用 100 个逻辑设备

sp_configure "number of devices", 100

2.38 如何为当前的 Sybase 数据库服务器设置可用的最大锁数？

Adaptive Server 中的所有进程共享一个公共的锁缓冲池，所需要的锁的数目取决于正在运行的操作的数目和性质。Number of locks 参数控制了 Adaptive Server 上进程可用的锁的数目。如果所有配置的锁都已经用尽，Adaptive Server 会显示一个服务器级的错误消息。另外，缓冲池锁会占用内存。

例如：允许最多使用 100,000个锁

sp_configure "number of locks", 100000

2.39 如何为当前的 Sybase 数据库服务器设置可用的最大用户连接数？

参数 number of user connections 用于设置可以同时连接到 Adaptive Server 的最大用户数。其中，用户连接包括用任何登录名（login）所做的登录连接，一个特定登录名（login）同时的多次登录被认为是多个连接。

例如：最多允许同时的 100 个用户连接

sp_configure "number ofuser connections", 100

2.40 如何备份数据库？

当遇到介质故障时（如磁盘老化损坏），仅当你有数据库的定期备份及后继的事务日志备份的情况下才可恢复该数据库。要完全恢复，必须定期使用 dump database 命令备份数据库并结合 dump transaction 命令备份日志。Dump database 命令允许动态执行，即转储/备份数据库期间允许其他用户对该数据库中的数据进行更改，这使得定期备份数据库更加方便。

示例：使用单一磁带设备进行转储/备份：

在 UNIX 操作系统中：

dump database pubs2 to "/dev/nrmt4"

在 Windows NT 操作系统中

dump database pubs2 to "\\.\tape0"

你也可以把数据库备份/转储到一个操作系统文件中。例如，在 NT 操作系统环境中：

dump database pubs2 to "d:\backups\backup1.dat"

2.41 如何进行数据库的增量备份？

可以使用 dump transaction 命令（或其缩写 dump tran）对事务日志进行例行的增量备份。Dump transaction 与许多操作系统提供的增量备份类似。它复制并且备份事务日志，提供自上一次数据库备份或者事务日志转储以来对数据库所进行的所有改变的记录。Dump transaction 复制/备份完日志后，会截断其中不活动的部分。

Dump transaction 比完全数据库备份所花费的时间和存储空间要少，一般更为常用。注意，只有将数据库的数据和日志分别存储在不同的数据库逻辑设备上时，才可以运行 dump transaction。示例：

在 UNIX 操作系统中：

dump transaction pubs2 to "/dev/nrmt4"

在 Windows NT 操作系统中

dump transaction to "\\.\tape0"

你也可以把数据库备份/转储到一个操作系统文件中。

例如，在 NT 操作系统环境中：

dump transaction pubs2 to "d:\backups\backup1.dat"

2.42 如何恢复数据库？

建议执行的步骤：

获取故障设备上每个数据库的当前日志转储（使用 dump transaction 的 no_truncate 选项）

检查设备上每个数据库的空间使用情况

收集完故障设备上所有数据库的这些信息后，删除每个受故障设备影响的数据库

删除故障设备

初始化新设备

重新创建数据库，每次创建一个

将数据库的备份装载到每个数据库

按事务日志转储创建时的顺序加载每个事务日志转储

online database YOUR_DB_NAME

2.43 如何截断数据库的事务日志？

事务日志填满数据库中的日志空间后，可能不能使用转储事务日志的办法备份并且清除原来存在的日志，因为转储日志这个动作本身也需要记录日志。

这时候，可以首先使用 dump transaction database_name with truncate_only 命令，该命令只是截断/清除事务日志，并不生成实际的备份。

如果不能奏效，可以使用 dump transaction *database_name* with no_log 命令。该命令也是仅仅清除既有的事务日志，不生成实际的备份文件，且该命令本身不记日志。

如果该命令还不能奏效，应当使用 alter database 命令为此数据库的日志分配额外的空间，随后执行 dump transaction。

2.44 如何恢复数据库到某一具体时间？

可以将数据库恢复到事务日志中的指定时间点。为此，使用 load transaction 的 until_time 选项。该功能在诸如用户不小心从数据库中删除了一个重要表时很有用；可用 util_time 选项将对包含此表的数据库的改变恢复到表被删除前的状态。

要在数据库破坏后有效使用 until_time，用户必须知道错误发生的确切时间，然后把包含误操作的事务日志转储出来，再如示例一样进行装载。示例：

```
load transaction database_name
from “/dev/nrmt5”
with until_time = “Mar 26 1997 12:35:59:650PM”
```

注意：这样装载事务日志后，AdaptiveServer重新启动数据库日志序列，在可以转储另一个事务日志之前，需要转储数据库。

2.45 如何把表、索引等数据库对象的定义从数据库中导出来？

使用 defncopy.exe 程序可以把指定的视图、规则、缺省值、触发器或者存储过程的定义从数据库导出到操作系统文件中，或者从操作系统文件导入到数据库中。它位于$SYBASE/OCS-12_5/bin 中。

示例 1：在 ABC 服务器上，将定义从 new_proc 文件拷入 stagedb 数据库中，使用 sa 帐户，口令为空：

defncopy –U sa -P –S ABC in new_proc stagedb

示例 2：在 SYBASE 服务器上，将 sp_calccomp和 sp_vacation 对象的定义从 employees数据库库拷入 dc.out 文件中，消息和提示以法语显示：

defncopy －*S SYBASE –U sa -P –z French out dc.out emploees sp* calccomp sp_vacation

2.46 如何把整个数据库中所有用户表的数据全部导出来？

bcp命令一次只可以把一个表的数据导出来。要想把所有用户表的数据都导出来，可以生成一个包含多行 bcp 命令的批处理文件，其中的每行 bcp 命令导出一个表。以下是生成这个批处理文件的示例：

a) 编辑一个文本文件 select.sql，内容如下：

```
set nocount on
use pubs2
go
select "bcp pubs2.." + name + " out " + name + ".bcp -Usa -P -c "
from sysobjects where type="U"
go
```

b) 如果是 unix 环境， 执行：

isql -Usa -P -b -i select.sql -o bcpout

chmod +x bcpout

如果是 Windows, 执行：

isql -Usa -P -b -i select.sql -o bcpout.bat

c) 把 bcpout 中 out 换为 in 即可得到 bcpin 的批处理命令文件

2.47 如何执行快速 bcp 操作？

当使用 bcp 命令向数据表中导入数据时，表上的索引和触发器会影响数据导入的速度。在这样的表上使用 bcp 时，bcp自动使用慢速模式，也就是将在日志中记录数据的插入，导致事务日志快速增长。

要想避免这种情况，可以先把目标表上的索引和触发器全部删除，并且把数据库 bulk copy/select into 开关打开，再使用 bcp 命令向该表中导入数据。

2.48 如何查看当前数据库中的进程信息？

可以使用存储过程 sp_who查看 Adaptive Server中的所有进程的信息。Sp_who 返回结果的解释如下：

fid　　进程所属于的组，在配置了并行处理的环境中有意义，否则为 0

spid　　进程号。系统管理员可以在 T－SQL 命令 kill 中用该号来停止进程

Status　进程状态，正在运行或者休眠

Loginname　启动进程的用户登录名或者别名，对于系统进程该项是 NULL

Origname　如果 loginname是别名，则 origname 显示真实登录名，否则与 loginname 显示同一信息。

Hostname　数据库驻留的服务器名称

Blk_spid　阻塞进程的 ID（如果有的话）。阻塞进程（可能受影响或者拥有排他锁）是占有其他进程所需资源的进程。

Dbname　该进程访问的数据库

Cmd　该进程正在执行的命令和进程。条件语句（if 或 while 循环）的求值将返回 cond。

Block_xloid　标识出阻塞事务的唯一的锁所有者 ID。

2.49 如何查看当前数据库中锁发生的情况？

可以使用存储过程 sp_lock查看 AdaptiveServer 当前运行状态中锁的分配和使用情况。

2.50 如何更改用户表的锁模式？

可以使用 alter table 命令来更改现有用户表的锁模式。

示例：把 titles 表的锁模式改为数据行锁：

alter table titles lock datarows

另外，一个表的锁模式可以使用 sp_help table_name 查看。

2.51 如何执行 optdiag 命令来监控表和索引的物理使用情况？

Optdiag程序用以显示数据库中关于表和字段的统计信息，也可以将更新的统计信息装载到系统表中。该程序位于$SYBASE\ASE-12_5\bin 中。

示例 1 ：显示 pubs2 数据库中所有用户表的统计信息，并将输出内容放入 pubs2.opt 中：

optdiag statistics pubs2 –Usa –Ppasswd –o pubs2.opt

示例 2：显示 titles 表的统计信息：

optdiag statistics pubs2..titles –Usa -Ppwsswd –o titles.opt

示例 3：显示 titles 表中 price 列的二进制统计信息：

optdiag binary statistics pubs2..titles.price – Usa –Ppasswd –o price.opt

2.52 如何执行 reorg 系列命令来优化 DoI表及其索引？

Reorg命令根据使用的选项，回收页上未使用的空间、删除行转移或将表中的所有行重新写入新页。通过整理表空间，可以有效改善表访问的性能。

可以使用的参数有：

reclaim_space **回收由删除和更新操作所留下的未使用空间。对于表中的各个数据页，如果有已经提交删除或者行缩短更新操作产生未使用空间，该参数将连续重写当前这些行，使未使用空间保留在页尾；如果页中已没有行，则释放该页。**

forwarded_rows **删除前移行**

compact **组合** reclaim_space **和** forwarded_rows **的功能**

Rebuild **如果指定了表名，则将表中的所有行重写到新的页；如果指定了索引名，**reorg**将重建所以。**

示例 1**：回收** titles **表中的未使用的页空间**

reorg reclaim_space titles

示例 2**：回收索引** titleind **中的未使用页空间**

reorg reclaim_space titles titleind

注意：reorg rebuild **要求将数据库选项** select into/bulkcopy/pllsort **设置为** true**，并在数据库中运行** checkpoint

2.53 如何使用 sp_sysmon 存储过程来查看当前数据库的性能情况？

存储过程 sp_sysmon **可以对指定时间段内** Adaptive Server **的活动情况进行十分详细的统计分析，并输出统计报告，系统管理员可以从输出报告中掌握到** Adaptive Server **的运行状况和配置参数的运行效果。输出报告的解读，请参阅性能和调优手册。**

示例 1**：统计未来** 10**分钟内** Adptive Server **的运行情况：**

sp_sysmon “00:10:00”

示例 2**：分别通过** begin_sample **和** end_sample **指定要分析的时间区间的其实端点和终结端点，其间可能发生了很多事务和操作。**

*sp_sysmon begin_sample ——**开始对运行数据采样***

…

*sp_sysmon end_sample ——**结束采样，输出统计报告***

3. 技术专刊

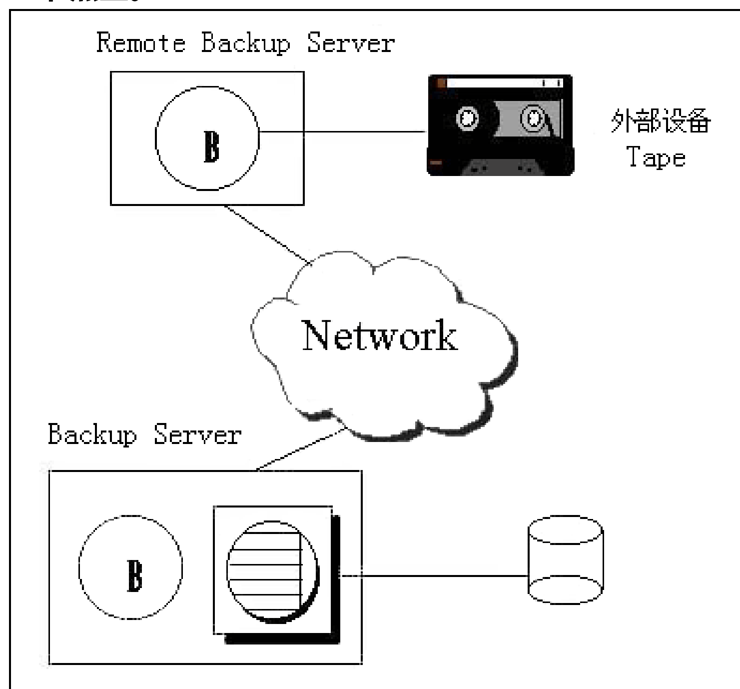
3.1 关于 Backup Server

什么是 Backup Server？

Backup Server 是一个基于 Open Server的工具。它可以处理 SQLServer10。0 以及更高版本的数据 Dump/Load。因为它是独立于 ASE 的处理过程，完成它的工作不依赖于 ASE。因此，在联机数据库 Dump/Load时，不会降低 ASE 的性能。

网络环境下的 Backup Server

Backup Server 必须与 ASE 放在同一台机器上（如果在 Open VMS 上，可以在同一簇内）。你也可以如下图所示的环境，使用分布在网络上的两个 Backup Server 来完成 Dump/Load。这两个 Backup Server 一个与 ASE 放在同一台机器上另一个可在远程某一节点上。



上图的配置，可以将本地（Local）的数据库 Dump 到远程（Remote）节点的外部设备上，或者从 Remote 节点读取 Dump，将数据库 Load 到本地的 ASE。

当将本地数据库 Dump 到远程机器的设备时，由 Local Backup Server 读取数据库数据，再送到网络另一端的 Remote Backup Server，经它将数据写到外部设备上。同样，从远程设备 Load 时，load Backup Server 发出命令给 Remote Backup Server，Remote Backup Server 读取 Dump 设备内容并传送到 Local Backup Server，由 Load Backup Server 将读到的数据写到数据库设备。网络上的 Dump 性能仅受网络传输能力的影响。

Backup Server 新的特性，提高了系统 Dump/Load的能力

- 能自动测出设备类型和密度；
- 支持多个 SYBASE Dump 到同一设备卷；
- 支持一个 SYBASE Dump 到多个设备卷。
-

Backup Server支持并行 Dump 设备

Backup Server支持最大 32 个 Backup设备并行地 Dump/Load。亦即可以将一个数据库分成碎片 Dump 到多个设备上。ASE 并行 Dump/Load的设备数受操作系统能打开的最大文件数和共享内存资源的限制。因此，各种不同硬件平台，不同的操作系统下，并行 Dump/Load 设备数不同。

多文件（Multi-file），多卷（Multi-volumn）Dump

- 操作系统文件和 Raw Partifion 是 Single_volumn，Single-file 的介质，这就是说这两种设备只能含有一个 SYBASE Dump。
- QIC 磁带和可移动硬盘是 Single-file,，Multi-Volumn介质，意味着一个 Dump可以跨越多个卷。这些设备不允许叠加，如果你想在同一个设备卷上追加存放多个 Dump，系统将提示是否覆盖已有的内容。
- DAT，8mm，9-Track磁带是 Multi-file，multi-volumn设备。以上三种设备可以包含一个或多个数据库 Dump。设备卷上的最后一个 Dump可以跨过磁带，继续存放在另一设备带上。

Backup Server 在网络上的 Server 名称

网络环境中的所有 Server 必须在 Interfaces 文件中有一个注册信息项，包括 Server 的名字，在网络中的网络节点地址及 SYBASE Server 用的网络端口号 Backup Server 也必须将它的信息放在 Interfaces 文件中。可以用 Sybinit 来完成这个工作。并且 Backup Server 的名字也必须加在master 数据库的 Sysservers 表中。如果你还有一个 Remote Backup Server，也需将它的信息放在本地的 Interfaces 文件中。

Backup Server 项在 Interfaces 文件和 Sysservers 表中必须有准确的名字。如果在启动 Backup Server 时用-s 选项指定了 Backup Server 的名字，那么在整个网络中都要用这个名字来标识这个 Backup Server。换言之，在 Interfaces 文件中，不能使用别名来标识某一个 Backup Server。

Backup Server 使用的设备

Backup Server 必须使用不回卷（No Rewind）设备，以便 Backup Server 能够控制磁带设备的位置。

下表列出了某些硬件平台上的 Dump 设备名称。其中 N 代表了设备号。

Platform	Type	Device Name
HP9000	4mm(SCSI and /HPIB)	/dev/rmt/Num
	9 track (/SCSI and /HPIB)	/dev/rmt/Nmn
NCR	8mm /SCSI (5 GB and 2.2 GB)	dev/nrmtN
	4mm/SCSI	dev/nrmtN
RS6000	9 track /SCSI	/dev/rmtN.1
	8mm DAT /SCSI (5 GB and 2.2 GB)	/dev/rmtN.1
	QIC /SCSI (1/4?cartridge	/dev/rmtN.1
Sun4	9 track /SCSI	/ dev /nrmtN
	QIC /SCSI (1/4?cartridge)	/ dev / nrarN
	8mm /SCSI (5 GB and 2.2 GB)	/ dev / nrstN
Sun4 SVR4 (Solaris)	9 track	See un4 SVR4?on page 4-10
	1/4 cartridge	
	8mm /SCSI (5 GB and 2..2 GB	
VAX VMS	9 track （HSC and /DSSI）	See pen VMS?on page 4-10
	8mm (/HSC 5 GB and 2 GB and /DSSI)	
	TK50 (HSC and /DSSI)	
AXP VMS	8mm / SCSI	See pen VMS?on page 4-10
	9 track	
	4mm DAT /SCSI	
AXP OSF/1	4mm DAT /SCSI	See XP OSF/1?on page 4-10
	TZ851	

Table 4-1：Device names for database dumps to tape

3.2 Backup Server的名字

当使用 Backup Server 做 Dump/Load时，系统会自动地寻找 SYB_BACKUP 这样的 Backup Server 名字。这就要求在 master 库的 sysservers 表中必须有一条记录来描述这个 Server，其 srvname 为 SYB_BACKUP。在安装 Backup Server 时，如果使用缺省值，那么安装过程将自动生成一个名字为 SYB_BACKUP 的 Backup Server，同时完成这些操作。

1.master 库的 sysservers 表中增加一条记录如下

srvid srvstatus srvname srvnetname

n 0 SYB_BACKUP SYB_BACKUP

2.$SYBASE/interfaces文件中增加一项，servername 为 SYB_BACKUP，有它自己的网络地址和网络端口号。

3.在$SYBASE/install目录下，创建启动 Backup Server 名为 RUN_SYB_BACKUP 的文件。

文件中启动 Backup Server命令的－S 选项指明 BackupServer在网上的名字为 SYB_BACKUP。

在做 Dump/Load时，系统自动地找 SYB_BACKUP 作为 BackupServer。再根据（1）中指出的 srvnetname 到 interfaces 文件中找到 server。但是，如果在安装时未使用 SYB_BACKUP 作为 Backup Server 名字。那么安装仅完成以上（2），（3）的操作。例如，你使用 MY_BACKUP 作为 Backup Server 名，则$SYBASE/interfaces 中 Server 名为 MY_BACKUP，启动 Backup Server 命令的－S 选项参数为 MY_BACKUP。完成（1）的操作，则必须依靠手工完成，在 ISQL 中，用 sa 帐号登录。

```
1>sp_addserver SYB_BACKUP, null, MY_BACKUP
2>go
```

其中 SYB_BACKUP 为固定参数，MY_BACKUP 是你指定的网络上 Backup Server 的名字。

3.3 Sybase 的数据库设备

数据库设备是指用于存取数据库的存储介质。它既可以是磁盘的一个分区，也可以是文件系统中的一个文件。但无论用哪种作数据库设备，它们在能被用来存取数据库对象之前必须初始化。一旦数据库设备被初始化，该设备就能：

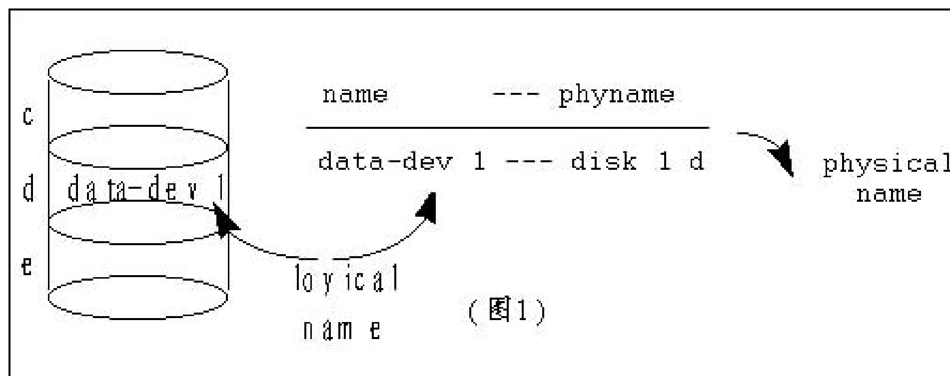
- 为某个用户数据库分配一块可用空间
- 在为某用户数据库分配的空间内指定并存储某个特定的数据库对象
- 用于存储某数据库的事务日志

选择磁盘分区还是 unix 文件来作数据库设备应视具体情况而定。一般来说，如果是最终用户的数据库，建议您用磁盘化区而非文件。因为对磁盘文件，ASE 不能确切

地知道数据是否真正被写到文件中去。如果是开发人员的数据库，那么用磁盘文件要稍好些，因为它使得建在其上的数据库恢复起来要容易一些。但对于 master device 来讲，无论是最终用户还是开发人员的数据库，磁盘文件无疑更为合适。因为相对而言，对 master device 的修改更少些而且镜像更为方便简单些。

如果在磁盘分区作数据库设备，应避免使用柱面 0,C,swap分区以及其它已被 OS 占用的分区。如果选择磁盘文件作数据库设备，要保证 SYBASE 用户在相关的目录下有写权限。

命令 diskinit 命令对数据库设备进行初始化，使得物理上的硬盘分区或文件可以被 ASE 使用，同时把逻辑设备映射到物理设备上，而每一次 disk init 执行后，都会在 sysdevices系统表中增加一行。其映射关系如图：



disk init的语法：

disk init
name = "device -name",
phyname = "physical-name",
vdevno = virtual - device - number ,
size = number - of - pages

其中：

name：	逻辑设备名
physname：	硬盘分区或磁盘文件名
vdevno：	虚拟设备号。它们取值可以是从 1到 max configured -1未被使用的任何值。可以用 sp_helpdevice或从 errorlog 文件中查看 device。用 sp_configure 来看 max configured的值。最大取值为 255。一般来说，the master device的 vdevno 总为 0，如果系统缺省设置最大设备数为 10，而您希望使用更多的逻辑设备，可以重新设置备数限制。

在 OS 下打入：

这是 sql 11.0.X 以下版本的配置

isql -Uxx -Pxx
1) sp_configure " device", 20
2) reconfigure with override
3) go

这个新的配置值必须在 Server 重新启动后才生效。若一个数据库设备被删除掉后，其 vdevno 只有在重新启动 Server 后才能被新的设备所用。

3.4 数据库选项（dboptions）的使用与管理

一、概述

ASE 能管理多个数据库，每个数据库有自己的选项。ASE 允许用户通过设置数据库选项，来更好地管理和使用数据库。当大批数据需要加载到数据库的表时，如果把该库的 Select into/bulkcopy 选项设置为 true，同时暂时删掉表上的索引和触发器，再使用系统提供的实用程序 bcp，数据将以惊人的高速度被装载到用户表里。也曾有用户抱怨，调用 dbwritetext 函数不能成功地把一幅 bmp 图写入数据库的 image 字段中。原因是用户数据库的 select into/bulkcopy 选项没有设置成 true。还有正在开发数据库应用的用户希望日常工作不受数据库日志满的不断打扰。这也可以通过设置数据库选项办到。因此，我们这里将数据库选项的设置和用法详细介绍给读者，希望它有助于你有效地使用和管理用户数据库。

二、数据库选项的设置

使用系统存贮过程 sp_dboption 来设置数据库选项，它的语法规则如下：

sp_dboption [dbname,optname,{true | false }]

其中：

- **dbname** 为用户所需设置选项的数据库名称。如果执行带参数的 **sp_dboption**，则当前使用的库必须是 master。不带参数便显示数据库选项清单。但是，用户不能设置 master 库的数据库选项。
- **optname** 为用户所要设置或关闭的选项名称。ASE 能识别选项名的任何唯一的字符串。当其名称为关键字或含嵌入空格或标点符号时，用引号括起来。
- **{true | false}**——设置该选项时为 **true**，关闭该选项时为 **false**。

使用 **sp_dboption**，能设置以下数据库选项：

- **abort tran on log is full**
- **allow null by default**
- **dbo use only**
- **ddl in tran**
- **disable free space acctg**
- **no chkpt on recovery**
- **read only**
- **select into/bulkcopy**
- **single user**
- **trunc log on chkpt**

查看数据库的选项设置情况，使用 **sp_helpdb** 存贮过程。

三、数据库选项的用途

（1）Selectinto/bulkcopy

使用 bcp 或批拷贝程序接口将数据高速拷贝到一张没有索引和触发器的表，Writetext 或者 Select into生成永久表，则此选项必须设置为 true。因此，当你使用 bcp 实用程序往用户数据库加载数据时，或者调用 dbwretetext函数把一幅图片插入到某张表的 image 字段，或者通过 Select into 命令形成结果表时，别忘了把该选项置为 true。ASE 为了快速执行上述操作，减少记录修改操作的日志。但是，这种不记日志的操作会使事务日志与数据库中数据不同步。这种事务日志对于系统失败后的恢复就没有用处。系统禁止在这种情况下，dump transation 到转储设备。因此，这些不记日志记录的操作完成后，将 Select into/bulkcopy选项关闭后，执行 dump database。

由 Select into 或批拷贝对数据库产生未被日志记录的修改后，执行 dump transaction命令时，显示错误信息，并指导用户改用 dump database（而 Wrrietext 命令没有这种保护）。

由于临时库 tempdb 从不恢复，所以使用 Select into 产生临时表时不必将 Selectinto/bulkcopy 选项打开。对于有索引的表进行 bcp 时，一般进行慢速批拷贝并且记入日志，所以此时也不必将 Select into/bulkcopy设置为 true。

(2)trunc log on chkpt

该选项表示每当 checkpoint 检查进程执行时（通常每分钟 12次以上），事务日志被载截，即将已提交的事务日志删除。但是，该选项设置成 true 以后，人工执行 checkpoint操作时都不载截事务日志。在开发数据库应用时，防止日志增长太快．可以将该选项设置成 true，所以它很有用。

（3） abort tran on log full

该选项决定当指定数据库日志段最后机会阀值被超越时，正在运行的事务的处理办法。缺省值为 false，也就是说事务被暂停挂起，直到空间空余出来再恢复。本选项设置为 true 时，在日志存储空间空余出来之前，所有用户的需要写事务日志的查询将被去掉。

（4） allow nulls by default

该选项置为 true，将列的缺省状态从 not null 改为 null，这与 ANSI标准一致。T-SQL 中列的缺省状态为 not null，即不允许列为空值。但用户可在列定义时指定该列为 null 状态。allow nulls by default true 与缺省状态相反。

（5） dbo use only

该选项设置为 true 时，只有数据库拥有者才能使用该数据库。

（6） ddl in tran

该选项设置为 true 时，用户在事务中可以使用某些数据库定义语言的命令，例如：Create table,grant和 alter table 命令。·只有 ddl in tran选项设置为 true 时，下述命令才可在用户自定义事务内部使用：

alter table create table drop rule create default create trgger drop table create index create view drop trigger create procedure ceate default drop view create rule drop index grant create Schera drop procedure revoke

下述命令在任何情形下都不能在用户自定义事务内部使用。

alter database load database truncate table create database load transation update statistics disk init recorfigure drop database select into

另外，系统过程 sp_helpdb，sp_helpdevice，sp_helpindex，sp_helpjoins，sp_helpserver 和 sp_spaceused，由于在执行时产生临时表，也不能在用户自定义事务中使用。再者，用户自定义事务中不能使用修改 master 数据库的系统过程。

（7）no free space acctg

该选项抑制剩余空间计帐及非日志段阈值活动的执行，由于不再计算这些字段的剩余空间，所以将加速恢复时间。关于该选项，在阈值管理部分，已有详细介绍。

（8）no chkpt on recovery

当保持最新的数据库备份时，此项设置为 true。此时，存在一个基本库和一个辅助库。首先，基本库转储到辅助库中。然后，基本库的事务日志间断地转储并装载到辅助库中

若此选项设置为缺省值 false，重新启动 ASE 时，恢复数据库后将向数据库增加一条检查点记录。检查点记录确保机制不会不必要地重新运行，改变顺序号，并导致随后从基本库装载事务日志的失败。

将辅助库的此项设置打开，恢复过程将不产生检查点，这样从基本库转储的事务日志可以被装载至辅助库。

（9）read only

该选项使得用户可以从数据库中检索数据，但不能修改数据。

（10）single user

该选项设置为 true 时，每次只能有一个用户访问该数据库。

3.5 Index & Performance

在应用系统中，尤其在联机事务处理系统中，对数据检索及处理速度已成为衡量应用系统成败标准。而采用索引来加快数据处理速度也成为广大数据库用户所接受的优化方法。

索引的使用效果不仅仅依赖于 ASE 的优化策略，在相当程度上也依赖于应用程序的设。怎样正确地使用索引，不能一概而论，究竟是让索引满足程序设计的需要，或是程序设计遵循已建立的索引，这两者是相符相承的。只有正确地使索引与程序结合起来，才能产生最佳的优化方案。

建立索引的目地是为了优化检索速度，如果检索所需要的时间过长，便有理由怀疑是否索引不存在或者优化器没有使用索引。尚若是索引不存在，那么就要取决于用户是否愿意用空间来换取时间，使用索引来解决检索速度慢的问题。如果优化器未使用表上已有的索引，那么要分析为什么，关于这一点将在后一点篇幅来说明如果 update 的效率很低，很可能是由于表上有太多的索引需要维护，从而浪费了时间。

优化器怎样使用索引

Table scan

如果表上没有任何索引，那么检索将采用 Table Scan方式进行，其所用时间主要依赖于表的大小。

例如：

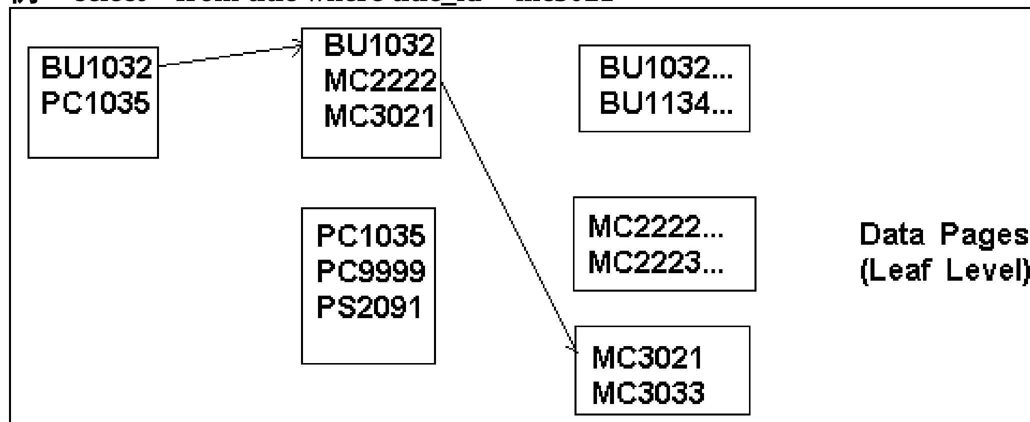
- dbcc checktable 测出表占 76923页
- -系统每秒读取 50 页
- -76923 页/50 页/秒=1538秒 （大约25分钟）

如果系统有比较大的 cache，某些数据可能由于以前已被读到内存，那么读取数据时间可能会低于估计的时间。一般情况下，Tablescan 检索是由于表上没有ClusteredIndex或者优化器认为，表中将有 20%的数据做为结果追回。

使用索引（条件为指定值）

索引中包含指定记录的值及地址，ASE 不必做全表扫描。

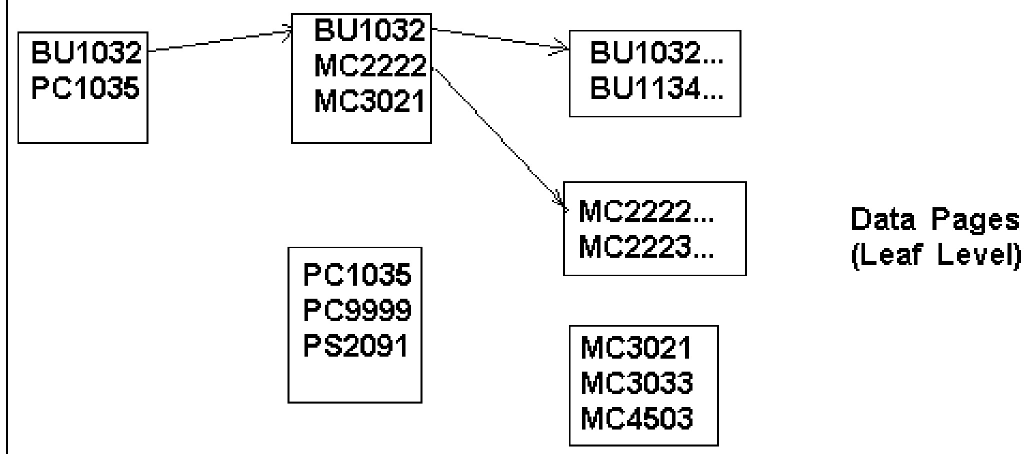
例：**select * from title where title_id="mc3021"**



当优化器认为读取索引页 I/O 加读取数据页 I/O 比做 Table Scan效果更好时，检索将使用索引。

使用索引（条件为某范围内值） 例：

```
select * from titles
where title_id >"BU1032"
and title-id <"mc3032"
```

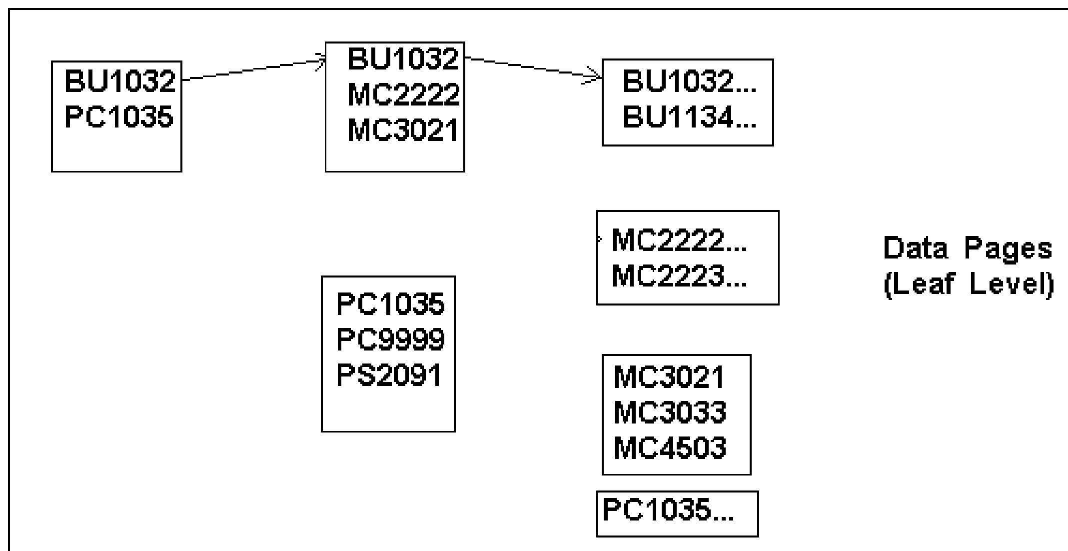


如果数据是排序的（有 Clustered索引），索引将被用来限制数据的扫描范围。

使用索引避免检索排序所需要的时间。

例：

```
select * from titles
order by title_id
```



对 Clustered索引来说，如果索引顺序与 Server顺序一致，那么上面的查询不需要重新排列返回结果。但是，若数据存储本身是升序排列，而查询要求降序排列，那么索引对加快查询没有任何作用。

对于 Non-Clustered索引，优化器将判断查询 Non_Clustered 索引页，找到满足条件的数据进行排序是否比 Table Scan 更快，优化器将找出最佳结果。从以上几例可以

看出，并非在表上建立了 Clustered 或 on-Clustered 索引之后，就一定会被使用，优化器是否使用索引取决于数据的查询命令，ASE 将从几个检索方案中选择最佳的一个。

在什么样的条件下才选择 Clustered索引呢？

选择什么样的索引基于用户对数据的检索条件，这些条件体现于 where 从句和 join 表达式。如果你的应用与以下情况相符，你可以考虑选择 Clustered 索引。

- 主键时常作为 where 子句的条件
- 某一列经常以这样的格式出现在 where 表达式中(x<=column <="y)"
- 某一列非常频繁地被访问
- 某列被用作 order by 或 group by
- 某列很少被改写
- 某列常出现在 join 中。

Non-Clustered 常被用在以下情况：

- 某列常用于 Aggregate 函数（如 Sum,....）
- 某列常用于 join,order by,group by。
- 查询检索出的数据不超过表中数据量的 20%

索引覆盖

索引覆盖是指 Non_Clustered索引项中包含查询所需要的全部信息。这种索引之所以比较快也正是因为索引页中包含了查询所必须的数据，不需去访问数据页。如果 Non-Clustered 索引中包含结果数据，那么它的检索速度将快于 Clustered索引。覆盖索引的缺点：由于索引项比较多，要占用比较大的空间。而且 update 操作会引起索引值改变。

ASE 对索引的限制

- 每个表上最多仅能有一个 Clustered索引。
- 如果表上有一个 Clustered索引，最多还能有 249 Non-Clustered 索引。
- 当没有 Clustered索引时，则可有 250个 Non-Clustered 索引。
- 索引最多建立在 256个列上。
- 当索引被创建时，ASE 需要 120%的附加空间。

索引维护

随着应用系统的运行，数据不断地发生变化，当数据变化达到某一个程度时将会影响到索引的使用。上面讲到，某些不合适的索引影响到 ASE 的性能，这时需要用户自己来维护索引。一种方法是删除老的索引，重新建新的索引。另外一种方法是保持索引统计有效（使用命令 update statistics），在以下情况下需要重新索引。

- 使用数据模式发生了较大变化。
- 某段时间内有极大量的数据插入。
- ASE 排序改变。
- dbcc 发现索引错误。

当重建 Clustered索引时，这张表的所有 Non-Clustered 索引将被重建。

维护索引统计表：

数据库拥有者必须用命令维护统计表。 update statistics table_name [index_name]

索引优化调整

用这条命令可以改善创建索引的性能,减少建索引所用的时间。 在 ASE 中可以调整如下参数：sp_configure "extent i/o buffers",nnnn 带来的影响是增加了 extent i/o buffers 大小，在 ASE 使用内存不变情况下，减少了 procedure 和 data cache，而且同一时刻仅有一个用户能用到 extent buffer。

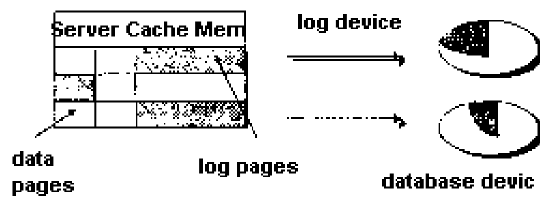
3.6 SYBASE ASE 事务日志

SYBASE ASE 的每一个数据库，无论是系统数据库（master,model, sybsystemprocs, tempdb），还是用户数据库，都有自己的 transaction log，每个库都有 syslogs 表。Log 记录用户对数据库修改的操作，所以如果不用命令清除， log 会一直增长直至占满空间。清除 log 可用 dump transaction 命令；或者开放数据库选项 trunc log on chkpt，数据库会每隔一段间隔自动清除 log。管理好数据库 log 是用户操作数据库必须考虑的一面。

下面就几个方面谈谈 log 及其管理：

一、ASE 如何记录及读取日志信息

我们知道，ASE 是先记 log 的机制。Server Cache Memory 中日志页总是先写于数据页：



Log pages 在 commit ,checkpoint,space needed 时写入硬盘。

Data pages 在 checkpoint,space needed 时写入硬盘。

系统在 recovery 时读每个 database 的 syslogs 表的信息，回退未完成的事务(transaction)（数据改变到事务前状态）；完成已提交的事务(transaction)（数据改变为事务提交后的状态）。在 Log 中记下 checkpoint 点。这样保证整个数据库系统的一致性和完整性。

二、Transaction logs 和 checkpoint 进程

checkpoint 命令的功能是强制所有"脏"页（自上次写入数据库设备后被更新过的页）写入数据库设备。自动的 checkpoint 间隔是由 ASE 根据系统活动和系统表 sysconfigures 中的恢复间隔（recovery interval）值计算出的。通过指定系统恢复所需的时间总量，恢复间隔决定了 checkpoint 的频率。

如果数据库开放 trunc log on chkpt 选项，则 ASE 在数据库系统执行 checkpoint时自动清除 log。但用户自己写入执行的 checkpoint命令并不清除 log，即使 trunc log on chkpt选项开放。只有在 trunc log on chkpt选项开放时，ASE 自动执行 checkpoint 动作，才能自动清除 log 。这个自动的 checkpoint 动作在 ASE 中的进程叫做 checkpoint 进程。当 trunc log on chkpt 选项开放时，checkpoint进程每隔 0秒左右清除 log，而不考虑 recovery interval 设置时间的间隔。

三、Transaction log 的大小

没有一个十分严格的和确切的方法来确定一个数据库的 log 应该给多大空间。对一个新建的数据库来说，log大小为整个数据库大小的 20%左右。因为 log记录对数据库的修改，如果修改的动作频繁，则 log的增长十分迅速。所以说 log空间大小依赖于用户是如何使用数据库的。 例如：

- update,insert 和 delete 的频率
- 每个 transaction 中数据的修改量
- ASE 系统参数 recovery interval 值
- log是否存到介质上用于数据库恢复

还有其它因素影响 log 大小，我们应该根据操作估计 log 大小，并间隔一个周期就对 log 进行备份和清除。

四、检测 log 的大小

若 log 在自己的设备上，dbcc checktable (syslogs) 有如下信息：

例:
```
***NOTICE:space used on the log segment is 12.87Mbytes,64.35%
***NOTICE:space free on the log segment is 7.13Mbytes,35.65%
```

根据 log 剩余空间比例来决定是否使用 dump transaction 命令来备份和清除 log。

用快速方法来判断 transaction log 满的程度。

```
1>use database_name
2>go
1>select data_pgs (8,doa mpg)
2>from sysindexes where id=8
3>go
Note:this query may be off by as many as 16 pages.
```

在 syslogs 表用 sp_spaceused 命令。

五、log 设备

一般来说，应该将一个数据库的 data 和 log 存放在不同的数据库设备上。这样做的好处：

- 可以单独地备份(back up)transaction log
- 防止数据库溢满
- 可以看到 log 空间的使用情况。[dbcc checktable (syslogs)]
- 可以镜像 log 设备

六、log 的清除

数据库的 log 是不断增长的，必须在它占满空间之前清除。前面已经讨论过，清除 log 可以开放数据库选项 trunc log on chkpt，使数据库系统每隔一段时间间隔自动清除 log，还可以执行命令 dump transaction 来清除 log.trunc log on chkpt 选项同 dump transaction with truncate_only 命令一样，只是清除 log 而不保留 log 到备份设备上。所以如果只想清除 log而不做备份，可以使用 trunc log on chkpt 选项及 dump transaction with truncate_only,dump transaction with no_log 命令。若想备份，应做 dump transaction database_na me to dumpdevice。

七、管理大的 transactions

有些操作是大批量地修改数据，log增长速度十分快，如：

- 大量数据修改
- 删除一个表的所有记录
- 基于子查询的数据插入
- 批量数据拷贝
-

下面讲述怎样使用这些 transaction 使 log 不至溢满：

大量数据修改　例 ：

```
1>update large_tab set col_1=0
2>go
```

若这个表很大，则此 update 动作在未完成之前就可能使 log 满，引起 1105 错误（log full）而且执行这种大的 transaction 所产生的 exclusive table lock，阻止其他用户在 update 期间修改这个表，这可能引起死锁。为避免这些情况，我们可以把这个大的 transaction分成几个小的 transactions，并执行 dump transaction 动作。

上述例子可以分成两个或多个小 transactions.

例如：

```
1>  update large_tab set col1=0
2>  where col2 <x
3>go
1>dump transaction database_name with truncate_only
2>go

1>update large_tab set col1=0
2>where col2>=x
3>go
```

```
1>dump transaction database_name with truncate_only
2>go
```

若这个 transaction 需要备份到介质上，则不用 with truncate_only 选项。若执 行 dump transaction with truncate_only，应该先做 dump database 命令。

删除一个表的所有记录：
例：

```
1>delete table large_tab
2>go
```

同样，把整个 table 的记录都删除，要记很多 log，我们可以用 truncate table 命令代替上述语句完成相同功能。

```
1>truncate table large_tab
2>go
```

这样，表中记录都删除了，而使用 truncate table 命令，log 只记录空间回收情况，而不是记录删除表中每一行的操作。

基于子查询的数据插入
例：

```
1>insert new_tab select col1,col2 from large_tab
2>go
```

同样的方法，对这个大的 transaction，我们应该处理为几个小的 transactions。

```
1>Insert new_tab
2>select col1,col2 from large_tab where col1<=y
3>go
1>dump transaction database_name with truncate_only
2>go
1>insert new_tab
2>select col1,col2 fro m large_tab where col1>y
3>go
1>dump database database_name with truncate_only
2>go
```

同样，若想保存 log 到介质上，则 dump transaction 后不加 with truncate_only 选项。若执行 dump transaction with truncate_only，应该先做 dump database 动作。

批量数据拷贝

在使用 bcp 把数据拷入数据库时，我们可以把这个大的 transaction 变成几个小的 transactions 处理，避免 log 剧增。

开放 trunc log on chkpt 选项

```
1>use master
2>go
1>sp_dboption database_name,trunc,true
```

```
2>go
1>use database_name
2>go
1>checkpoint
2>go
bcp... -b 100 (on unix)
bcp... /batch_size=100(on vms)
```

关闭 trunc log on chkpt 选项，并 dump database。

在这个例子中，一个批执行 100 行拷贝。也可以将 bcp 输入文件分成两或多个分开的文件，在每个文件执行后做 dump transaction 来避免 log 满。

若 bcp 使用快速方式（无索引，无 triggers），这样操作不记 log，换句话说，log 只记载空间分配情况。在这种情况下，要先做 dump database（为恢复数据库用）。若 log 太小，可置 trunc log on chkpt 选项，这样在每次 checkpoint 后清除 log。

八、Threshold 和 transaction log 管理

ASE 提供阈值管理功能，它能帮助用户自动监视数据库 log 设备段的自由空间。这方面的详细讨论见 NO.5 技术支持杂志。log 的管理是灵活而复杂的，我们应该在实践中摸索经验，针对每个数据库的不同情况，不同操作，做不同处理。

3.7 阀值管理（Threshold Management）

在使用 Sybase 数据库管理系统（ASE）开发企业应用系统时，或者开发好的数据库应用系统投入实际运行后，由于用户不断地增加或者修改数据库中的数据，用户数据库的自由存储空间会日益减少。特别是数据库日志，增长速度很快。一旦自由空间用尽，ASE 在缺省情况下挂起所有数据操纵事务，客户端应用程序停止执行。这样有可能会影响企业日常业务处理流程。Sybase ASE 提供自动监视数据库自由存贮空间的机制——阈值管理，当数据库使用剩余空间低于一定值时，通过执行一个自定义的存储过程，来控制自由空间。在空间用完之前，采取相应措施，这样有利于业务处理顺利进行。如果能充分利用 ASE 的阈值管理功能，用户能使一些数据库管理工作自动化，规程化。所以，在此我们将 ASE 这一重要功能介绍给读者。

ASE 的阈值管理允许用户为数据库的某个段上的自由空间设置阈值和定义相应的存储过程。当该段上的自由空间低于所置阈值时，ASE 自动运行相应存储过程。与阈值相对应的存储过程由用户定义，ASE 不提供。一般来说，数据库管理员可通过这些存储过程来完成一些日常管理事务，例如：

备份数据库，清理日志和删除旧数据
备份数据库日志
扩展数据库空间
拷贝出表中的内容，清理日志，等等。

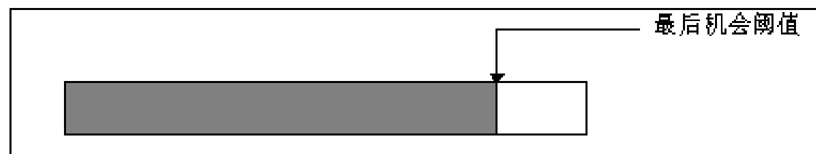
（一）段（Segment）

ASE 的阈值管理是基于段（Segment）的，因此，让我们先回顾一下段的概念。每个数据库创建时，它有三个缺值段：（1）System 段；（2）default 段；（3）logsegment 段。以后，还可以为该数据库增加用户自定义段。将来所有的数据库对象

都建立在这些段上，要么是系统定义的段，要么是用户定义的段。数据库的系统表存放在 System 段上。在没有指明段时，建立的对象存放在 default 段上。数据库的事务日志放在 logsegment 段上，该段是通过使用建立数据库（Create database）命令的 log on 选项来定义的。

（二）最后机会阈值（Last_chance Threshold）

缺省情况下，SQLServer 监测日志段的自由空间，当自由空间量低于事务日志能成功转储的需要时，ASE 运行 sp_ thresholdaction 过程。此自由空间量称为最后机会阈值（Last_chance threshold），它由 ASE 计算得来，并且用户不能改变。



sp_thresholdaction 必须由用户编写，ASE 不提供。另外，如果最后机会阈值越出，那么在日志空间释放前，ASE 一直挂起所有事务。但可以使用 sp_dboption 对某一数据库来改变这一行为。设置 abort tran on log full 选项为 true，可使得最后机会越出时，ASE 撤回所有还未被注册的事务。

（三）阈值管理

系统缺省建立最后机会阈值，由用户编写缺省阈值处理存储过程（sp_thresholdaction），来控制自由空间。除此之外，还可以使用以下存储过程管理阈值：

- sp_addthreshold 建立一个阈值
- sp_dropthreshold 删除一个阈值
- sp_helpthreshold 显示阈值有关的信息
- sp_dboption 改变阈值的“挂起或取消”行为和取消阈值管理
- sp_helpsegment 显示某个段上空间大小和自由空间大小的信息

（四）增加阈值（sp_addthreshold）

它用于创建阈值（threshold）来监测数据库段中空间的使用。如果段中自由空间低于指定值，ASE 运行有关的存储过程。增加阈值的命令语法为：

sp_addthreshold database, segment, free_pages, procedure

其中：

- database——要添加阈值的数据库名。必须是当前数据库名称。
- segment——其自由空间被监测的段。当指定“default“段时要用引号。
- free_pages——阈值所指的自由空间页数。当段中自由空间低于该标准时，ASE 运行有关存储过程。

- procedure——当 segment 中的自由空间低于 free_pages 时，ASE 执行该存储过程。该过程可以放置在当前 ASE 或 Open server 的任意数据库中。但是，超出阈值时，不能执行远程 ASE 上的存储过程。

例如：sp_addthreshold pubs2, logsegment, 200, dump_transaction

其中，存储过程定义为：

```
create procedure dump_transaction
@ dbname varchar (30),
@ segmentname varchar (30),
@ space_left int,
@ status int
as
dump transaction @dbname to "/dev/rmtx"
```

那么，当日志段上可用空间小于 200 页时，ASE 运行存储过程 dump_transaction, 将 pubs2 数据库的日志转储到另一台设置上。

sp_addthreshold 不检查存储过程存在已否。但当阈值越出时，如果存储过程不存在，ASE 把错误信息送到错误日志（ errorlog ）中。系统允许每个数据库有 256 个阈值，而同一段上二个阈值之间的最小空间为 128 页。其存储过程可以是系统存储过程，也可是在其它数据库里的存储过程，或者 Open Server远过程调用。

（五）删除阈值（sp_dropthreshold）

它删除某个段的自由空间阈值，但是不能删除日志段的最后机会阈植。删除阈值的命令语法为：

```
sp_dropthreshold database_name,segment_name,free_pages
```

其中三个参数分别为：阈值所属数据库名，阈值所监测的自由空间的段名，和自由空间页。例如：

```
sp_dropthreshold pubs2, logsegment, 200
```

删除 pubs2 库中 logsegment 段的阈值 200。

（六）显示阈值（sp_helpthreshold）

它报告当前数据库上与所有阈值有关的段，自由空间值，状态以及存储过程或报告某一特定段的所有阈值。显示阈值的语法为：

```
sp_helpthreshold [segment_name]
```

其中 segment_name 是当前数据库上一个段的名字。

例如：

- sp_helpthreshold logsegment **显示在日志段上的所有阈值**
- sp_helpthreshold **显示当前数据库上所有段上的全部阈值**

（七）sp_dboption的新选择

abort tran when log is full

当日志段的最后机会阈值被超越时，试图往该日志段上记日志的用户进程将被挂起还是被撤回？缺省情况下系统挂起所有进程。但是可以使用 sp_aboption **改变它。执行** sp_dboption salesdb,"abort tran when log is full",true **命令后，一旦日志满了，则数据库修改事务将会被回滚。**

disable free space acctg

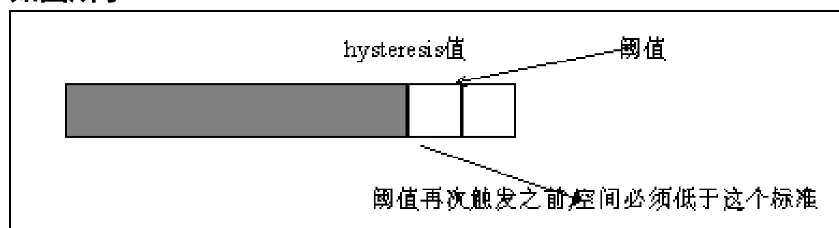
这个选择取消数据库中的阈值管理，但不影响最后机会阈值。执行：sp_dboption saledb,"disable free space acctg",true **它取消对非日志段上自由空间的统计。取消后，对系统有以下影响：**

- ASE **仅计算日志段上的自由空间**
- **日志段上的阈值继续处于活动状态**
- **在数据段上，系统表不改变，并且** sp_spaceused**得到的值是该选择被取消时刻的值**
- **数据库段上的阈值失效**
- **恢复加快**

（八）阈值的触发过程

频繁的插入和删除可能会使数据库段中的空间波动，阈值可能被多次超越，ASE **使用系统变量**@@thresh_hysteresis**，避免连续触发阈值存储过程。它的值由** ASE **设定。在** system 10.0**中，**@@thresh_hysteresis **是** 64**页。**

如图所示：



因此，增加一个阈值，它必须与下一个最近的阈值相距至少 2*@@thresh_hysteresis **页。**

一个阈值被触发，需要以下几个条件成立：

- **必须到达阈值**
- **阈值处于活动状态（即它被建立后或者自由空间达到阈值减于** @@thresh_hysteresis**）**
- **只有自由空间减少阈值才触发，如果自由空间增加，它永远不触发**

3.8 日常后备数据库

SYBASE 数据库的备份主要是通过 dump 命令来实现的， 分为数据库备份和日志备份。 简单语法格式如下：

```
dump database database_name to device_name
dump transaction database_name to device_name
```

如果使用磁带设备做备份, 还要使用 capa city、 init 选项。

应该多长时间备份一次数据库呢? 这决定于发生系统故障时(例如磁盘故障) 应用系统允许丢失多长时间的数据。 如果允许丢失一个小时的数据, 那么可以考虑每天至少备份一次数据库, 并每个小时至少备份一次日志。

备份数据库之前, 应对数据库做 dbcc 检查, 确保备份的数据库是完好的数据库。如果数据库有损坏, 备份时可能不会报错, 但将来可能无法装载（load）。

通常情况下, 只有发生严重故障需要恢复时才进行装载数据库, 但是 SYBASE 建议偶尔对备份的数据库装载到一个测试环境上, 以保证备份工作过程正常并熟悉备份和装载工作过程。

Master 数据库的备份也不能忽略。每次 master 数据库的内容有变动时, 应及时备份 master 数据库, 例如增加 login, 增加设备, 增加用户数据库等。

用户可根据自己的应用制定可行的备份方案。 详细的命令语法及备份方案的详细建议均可在 SYBASE 手册中查到。

举例：Unix 平台后备 pubs2数据库

- 启动 ASE 和 BACKUP SERVER
- isql -Usa -Pxxxxxx -SSYBASE

  dump database pubs2 to "/sybase/dump/pubs2_db.990705"

  go

  dump transaction pubs2 to "/dev/nrct0" capacity=1500000，file="pubs2_log.990705"

  with init(capacity 取值为磁带容量的 70% 左右,以 K 为单位)

  go

3.9 使用 dump transaction with no_log 的危险性

在命令参考手册中的 dump transaction with no_log 条目下，有一条警告信息告诉你，你应该把这条命令作为没有其它办法时的最后一招才使用它。但是"最后一招"究竟是什么意思呢？当你使用这条命令时会怎样呢？那你应使用哪条命令来代替它呢？最后，若这条命令如此有问题，为什么 Sybase 却要提供它呢？

Sybase 技术支持建议你定期的 dump 你的 transaction log。你必须根据你的数据库中记入日志的活动量的大小以及你的数据库的大小来决定 dump 的方式。有些地方按月 dump transaction；有些地方每夜 dump transaction。

若你从未做过 dump transaction，transaction log 将最终会满。 SQLServer 使用 log（日志）是出于恢复目的的。 当 log 满时，服务器将停止事物的继续进行，因为服务器将不能将这些事物写进日志，而服务器不能运行大多数的 dump tran 命令，因为 ASE 也需在日志中记录这些命令。

这就是为什么当其它 dump tran 命令不能执行时 no_log 可执行的原因。但是想一下 dump transaction with no_log 被设计执行的环境，将不做并发性检查。

若你在对数据库的修改发生时使用 dump transaction with no_log，你就会冒整个数据库崩溃的风险。在多数情况下，它们被反映成 813 或 605 错误。为了在数据库被修改时，删除 transaction log 中的不活跃部分可使用 dump transaction with truncate_only。这条命令写进 transaction log 时，并且它还做必要的并发性检查。这两条命令都有与其相关的警告，在命令参考手册中会看到这些警告。请确保在使用其中任一条命令以前，你已理解这些警告和指示。 Syase 提供 dump transaction with no_log 来处理某些非常紧迫的情况。为了尽量确保你的数据库的一致性，你应将其作为"最后一招"。

3.10 在 ASE 11.9.2 版中采用了行级加锁机制以提高性能

Sybase 公司关系型数据库管理系统产品的最新版本----Adaptive Server® Enterprise 11.9.2 通过对行级加锁机制的创新以满足日益增长的可伸缩性的需求，并使得产品的性能和可伸缩性得到了明显的提高；特别是在商业应用中，使得这个产品能够满足尽量少占用硬件和操作系统资源的用户需求。本白皮书对这种新加锁机制的实现方法进行了详细的介绍，并且针对特定的采用页面级加锁机制的应用系统和以前所推荐的提高性能的典型优化方法，就新的行加锁机制所带来的效益进行了量化的分析。

随着 Adaptive Server Enterprise (ASE) 11.9.2 的发布，Sybase 公司已经使得其数据库管理系统产品 ASE 在 11.9 版本中所申明增加的功能都得到了实现。这次发布 ASE11.9.2 表明了对于那些采用较早地使用了 Sybase ASE 产品的应用系统，它们所遇到的一系列与并发处理有关的课题得到了解决。

从高层的观点看来，有人可能会说新功能不过只是实现了行加锁功能。然而实际上却不止如此，Sybase 公司已经抓住了这次机遇，在新产品中实现了包括新型存贮和索引等在内的范围广泛的新技术，以扩大其产品的功能并改善产品的性能。这种方法不仅是提供了更小尺度的加锁机制，而且还比其它一些方法诸如传统地对索引、分配、OAM,和 GAM 页面加锁进行限制明显地提高了性能。

在本技术报告文件中，通过一些特定的实例对新的加锁机制进行了阐述，而这些实例都是使其并发性及其性能得到了很大的提高。应该注意到，为了使本文件内容更加集中，所提供的材料都是在很高层次上的。有关更详细的信息，请参看如下的文件：

- **New Functionality in Sybase Adaptive Server Enterprise 11.9.2[1]，本文件是** ASE 11.9.2 **文档系列之一，该资料详尽介绍了本版本中所包含的全部新功能。**
- **When (and When not to) Use Data-Only Locking [2]（ by Ian Smart），在这本由笔者撰写的资料中包含了本文献中所引用实例的详细分析、测试数据和硬件/软件配置。**

3.10.1 所支持的加锁机制

1) 全页加锁

全页加锁既是一个新术语，它又是由 ASE（Adaptive Server Enterprise）**在过去所支持的一种加锁类型。这种类型有下列特性：**

- **对所有可被访问的页面在页面级加锁**
- **当各种类型的页面以任何方式发生改变时，对这些排它性的页面进行加锁；而且这种加锁机制一直保持到该事务终止；**
- **当下一个所需的页面已经成功地获得，对那些已经释放的的当前访问页进行共享页面加锁(如果采用了第三层** ANSI **隔离，则把这种加锁机制保持到该事务终止为止)**
- **采用页级时间印记（**timestamp**）以确定是否发生改变，详细信息记录在事务日志中，以便在系统恢复时以向前或向后方式使用。**

这种加锁方式常常提供性能最高的解决方案，特别是当应用设计时已经考虑了这些特性时更是如此。但是，有一些应用系统，当发生某些活动时，这种对整个页面进行加锁的方式就可能会对系统性能产生有重大意义的影响。对于那些面对诸如文件系统或其它已经支持更细小尺度加锁机制的数据库厂家产品的一般环境而设计的应用系统而言，这种情况尤其如此。

此外，还存在一系列问题，它们要围绕着更加困难的条件进行工作。它们通常要采用更加具有 Sybase **特性的解决方案。对于商用的应用软件制造厂商而言，对他们是一个挑战，因为这将要求他们必须跨越他们所支持的数据库平台，去完成维护其原代码的工作，而这个工作有相当工作量。在这个领域的基本问题如下：**

- **对已经按照升序值创建的非群聚性索引的最末端叶型页面存在着争议**
- **对非群聚性索引的表进行插入和查询时可能发生死锁；**
- **在按照群聚性的索引值进行更新和对非群聚性索引的表进行查询访问之间可能发生死锁；**
- **在没有作索引的表的最后一行可能发生冲突(尽管对最后的特定地址可以使用分区)；**
- **有可能使行数很少的表之间发生潜在的冲突(尽管对特定的地址可以使用填充因子**[fillfactors]**和每页最大行数**[max_rows_per_page]**这两个参数**)
- **对每个页面两边进行加锁的需要常常被分割开来；**

- 如果一个表特别小，以致在一个单一页面中进行驻留，那么对单一行的访问实际上将破坏对整个表的加锁机制。

2) 仅对数据加锁

仅对数据加锁机制试图去解决本文前一节所关注的主要问题(其他的议题将在其它功能领域中加以解决)。这种加锁方式支持两类不同的工作方式： 数据行加锁和数据页加锁。在这两种情况中，对于它们所支持的加锁方式，都与以前的加锁机制有所不同。仅对数据加锁具有下列特性：

- 在索引页面中不会破坏事务加锁。相反，而是采用了一种称之为锁存的机制。锁存是一种类似于旋转锁（spinlocks）的同步方法，它们与事务无关并且只保留很短的周期（一般而言，当一个任务在数据库中物理上改变一小片数据时，这个周期相当于在共享存贮区中在一个 2K 的页面改变某些字节数据的时间）一旦完成之后，这个任务将直接打开这个锁存。当这种情况还可能临时同其它组块时 ，因为这种锁存不能对服务器任务进行有上下文的切换，也不能涉及死锁，并且只能保持主要的一小段时间，所以它们不能产生有显著意义的争用。
- 采用一个 RID 对单一行进行数据行加锁(行标识[RID----Row ID]是逻辑页号与所在页面上该行号的组合)；
- 支持固定的行标识 RIDs, 它可以是向前的，允许不进行其 RID 的改变，就完成数据行的移动。当一行变大超过了它的可用空间时，采用上述结果对非群聚索引不需要进行任何改变。
- 不需要进行任何争用就可以在表的尾部进行插入操作，这一功能已经增加进来。.
- 支持采用范围加锁、下一个关键字加锁和无限大加锁等方式对逻辑范围值进行加锁
- 支持由最顶层操作所导致的页面分割。这些情况直接加以提交，"系统"事务可以导致在更短一点的时间周期里保持分裂的页面处于锁定状态。

为了支持这些变化，需要对采用的存贮表结构进行一系列改善。这些改进的主要效果如下：

- 群聚索引现在被存贮为象许多人所熟悉的 IBM DB2 产品所采用的“放置索引”（"placement indexes."）方式。这种结构类似于非群聚性的索引，需要类似的空间总量。这种修正的结构导致了在数据初始存贮时可以按照顺序跨数据页进行存储，但是当发生插入时，它们就要尽可能紧密存放以便在正确的逻辑页面中不存在页面分割。此外，在数据页中的数据顺序在新行增加时是不进行维护整理的。这种索引的应用使每个群聚化的索引周游增加了一次 I/O 操作。

- 行位移表已经增加到索引页和数据页中。这种增加和新的行索引行存贮格式具有使每个索引页面所存贮的索引条目个数减少的潜在能力。
- 固定行标识（RIDS）。当一行移动时，对于分配新行位置的向前地址被放在用于驻留该行的位置上。当这种移动需要改变非群聚性索引时，对该行的访问需要增加一次 I/O 操作以得到‘向前’的位置。
- 一般而言，索引将更小和更短，这是因为如下原因：

  - 从每个叶级页面中采用双重键限制机制来限制双重键（Duplicate key）例如，如果值“绿”（"GREEN”）在下列行标识（RIDs）值等于 123-1，234-2，和 345-3 的行中, 就分别存贮值“绿”（"GREEN”），123-1，234-2，345-3，而不是存贮值“绿”（"GREEN,”）三次。在每个索引页中每个值只存贮一次。
  - 在非群聚性索引树的非叶型结点中将后缀实行压缩(例如，如果键值是"GREEN”和"HAMILTON”，而在这两个值之间发生分裂，那么就在非页级索引页面中存储"G’和"H”)。

3) 数据页和数据行加锁

只对数据加锁机制支持两种方式：数据页加锁和数据行加锁。这些与它们的工作方式和所提供的功能相类似。这两种方式仅在对数据访问产生阻碍作用时，在加锁的尺度上有所区别。在数据页加锁方式下再采用数据行加锁方式具有两种作用（一种起正向作用，另一种起反向作用）。首先，较小尺度加锁机制的使用可能导致减少争用与冲突，然而当大量数据发生变化时，就有可能对加锁产生大量阻碍的情况发生。

4）特定使用的加锁类型

除非对配置参数加以特定，对所有的表都予置了隐含的全页面加锁机制。

sp_configure ‘lock scheme’,0, [allpages | datapages | dat arows]

当数据库从原先版本的服务器中转储出来重新加载时，所有的表都被定义为全页面加锁的表。当建立一个新表时，可以不使用这个缺省值，可采用如下的句法格式：

create table <tablename>…lock [allpages | datapages | datarows]

为了在使用的一个表中改变加锁类型，可以采用如下的句法格式：

alter table <tablename> lock [allpages | datapages | datarows]

在一个现存的表中改变加锁方式，将引起下列三种行动后果发生：

- 首先，如果一张表从全页加锁转变为仅对数据加锁，或者从仅对数据加锁转变为全页加锁，在这两种类型之间就要对表进行选择以允许进行存贮格式改变。如果这是一个分区表，就要同时假定必要的并行级别和工作线程已经配置好的情况下，才能执行。

- 其次, 对表中的群聚性索引必须重新创建。因为我们能保证数据，所以如果从全页加锁方式转换为只对数据加锁时，这种重新创建可以通过"with sorted_data"来完成。然而，当从仅对数据加锁机制转换为全页加锁方式时，就要进行并行的索引创建操作。（请注意：如果这是一个分区表时，那么并行等级和工作线程的数目必须加以配置才能允许进行这种改变，否则这种迁移将会失败）
- 最后，非群聚性的索引将被重建，如果服务器已经为并行处理所加以配置，当进行本步骤时将加以采用。

由于这些活动同潜在的工作量有关，从全页加锁机制改变为仅对数据加锁或从仅对数据加锁改变为全页加锁机制都可能是耗费时间的活动。为了标注这一点，有以下一些选择：

- 如果可能的话，应该配置使用并行方式。这至少对执行非群聚性索引的哈斯（杂凑，即 hashed）创建方法是必须的，但是如果可能的话，采用分区表和分区扫描将使系统得到更大的改进。
- 在选择进入和创建群聚性的索引之后，该任务将被设置检查点（checkpointed ）所以，如果有充分的硬件资源，通过允许在任何一个时间点上，检查点任务可以具有多于 10 个（系统缺省值）的异步 I/O 请求，利用 dbcc 进行调谐将能够带来有益的效果。（'maxwritedes', number)
- 进一步作为降低使用检查点成本的一种方法，在相关的高速缓冲池（cache pool）、大数据量的 I/O 操作中，采用对高淘汰程度进行标记的方法，并允许清洁程序（好象家庭主妇一样）保持特别活跃的状态，将为那些检查点需要从高速缓冲池中刷新较"脏"的页面的而增加的 I/O 操作次数，并因此花费了在检查点上的时间，都能够大大减少作出贡献。
- 如果预先进行了配置，则可以对并行的选择进入可以使用预先分配的盘区。所以，通过将 sp_configure number of pre-allocated extents 设置为 16 也将对系统性能有明显的积极的效果。

备注：在仅对数据加锁类型之间进行改变不需要对数据进行备份，而且执行起来只需很短的一段时间。

3.10.2 对小量数据的多次并发访问

1) 在小型表中采用不同加锁方式之间的比较

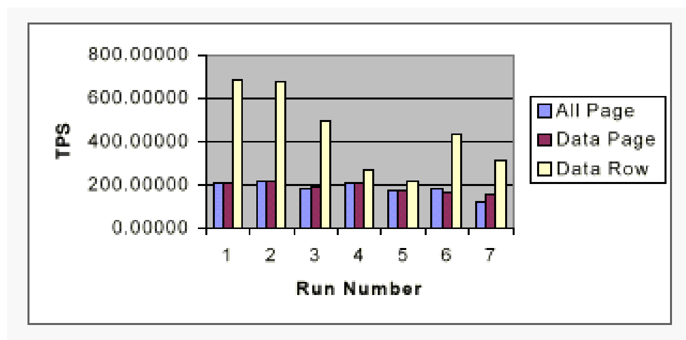
对于那些被访问的数据同时（或者）仅有一张或很少数量的页面发生改变的情况，在页面级进行加锁极有可能导致表中很大比例的页面都实施了加锁机制。使用按数据行进行加锁将因为降低了加锁的尺度而明显增加了系统的吞吐量。

为了对使用按数据行进行加锁改变了原先发生的组块移动，已经设计了一系列基于 TPC/B 基准的测试实验。在这些实验中，使用了各种 TPC/B 的变量以创建略有不同的条件和可能发生的情况。在下面这些结果中，前三个执行的是直接更新。在实验 4 与 5 中，采用了选择、删除以及插入的组合（请注意，这些实验比其它的实验作了更多的工作）在实验 6 与 7 中， 由于补偿索引直至补偿列引起了数据更新的拖延。

利用已经执行的全部工作，在更新三张表的实验中，最小的表总是最后一个被更新。然后事务在历史记录表中插入一个条目。

备注：下列测试是在具有 12 个处理器的 HP9000/891 T500 的设备谁进行的。Adaptive Server Enterprise 配置为具有 10 个驱动核心（ engines）和 1GB 的内存。

测试结果如下图所示：

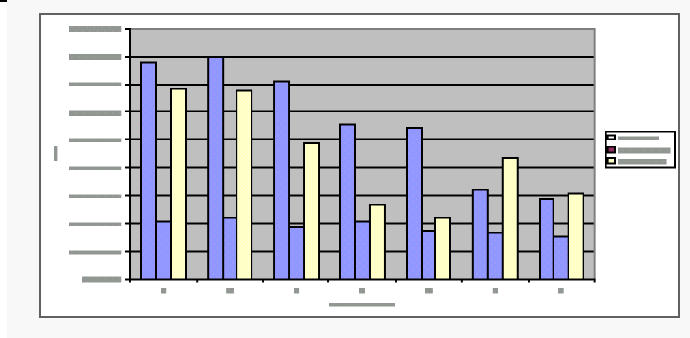


从这些数据可以看出，利用数据页加锁方式与全页加锁方式相比较而言，对于吞吐量来说几乎没有什么差别。这是由于两者之间在加锁尺度方面没有什么差别。然而，一旦使用了数据按行加锁方式，100 行的数据可以单独访问时，此时在性能上的明显改进就能够达到了。

2) 在小型表中采用填充因子（Fillfactor）与仅对数据加锁方式之间的比较

在上文中所提及的小型表中，对表传统的工作方式是采用了诸如在每页的某一行，在群聚性索引上有效地放置了数目为 1 的填充因子。我们采用 100 行的表重复进行了同样的一组实验，此处表都已经具有采用这类填充因子创建的群聚性索引。

下图显示了实验结果，它们分别是较早时间进行的按数据页和数据行加锁的结果，以及与之形成对照的采用 1%的填充因子的全页加锁实验结果：



正如所看到的，在直接更新的情况，对吞吐量来说有一点微小的提高；而在删除/插入的测试情况中，对吞吐量却有明显的提高；而对延迟的更新实验，吞吐量则要有一些降低。

这些实验说明对这些课题来说，Sybase公司传统的响应在大多数情况下还是有效的，而且对那些已经用这种方法实现的用户来说并没有明显的缺陷。然而，对于填充因子方法来说还是有一些缺点，因而就限制了它们的用途。

3) 何时采用只对数据加锁或填充因子

随着引入仅对数据加锁方式，现在数据库设计者对使用更新的表中，如何解决争用冲突课题方面，有了选择。在表比较小或保持较小时，采用填充因子对于不使用延迟更新的情况将对吞吐量的提高有轻微的改善作用。

如果表比“少数”页大，那么填充因子的使用，特别是该因子为1时，就成为应该禁止和代价昂贵的一种方式了。这是由精确度所决定的，填充因子在每个页面里都留下了一些没有使用的空间。这样一来明显增加了磁盘空间需求，对大表尤其如此，这将导致更大的数据库和对更多硬件的需要。

其次，因为填充因子不是永久的，客户必须安排重构索引的程序段以对性能进行调整和维护。这样一来使争用和冲突发生在特定的区域，使之变得更加令人烦恼。事实上在过去，当用户激活逻辑分区的数据，静态数据放在另外的表里时，就把静态数据紧紧地包装在一起，在利用填充因子使之激活，这种情况对于用户并不是罕见的情况。使用这种技术已经引用了把数据放回到一个单一的结果集合中来的组合操作，这使得编写SQL程序变得格外复杂。

采用仅对数据加锁的方法可以使用户不一定需要使用填充因子来解决更好地利用可用空间的问题。这种选择使得用户可以在“两个世界之间的最佳者”选项中作出决定，以为其环境选择最优的方案。

3.10.3 堆栈表和热点

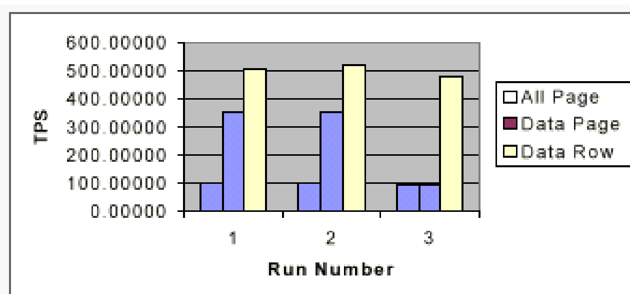
1) 在堆栈表中增加行

应用系统更新行时，如果同时使用了填充因子（fillfactor）或每页最大行数（max_rows_per_page），在这一话题是插入一行时，特别是在特定的位置（即热点--"hot spot"）插入时，这些方法就变得几乎没有价值。无论何时在一个没有群聚性索引的表（即所谓"堆"表–"heap"）中进行插入时，这种插入总是插到表的最后一页中去。在 Sybase ASE 11.0之中，Sybase 公司提供了允许用户在分区表中把这种插入蔓延分散到跨分区里去。然而，在采用这种方式解决问题的同时，这种方法不能处理由于空间开销而不能使用堆的情况，也不能解决那些因为查询原因要把数据按着添加的顺序存放的表的问题。

随着在 Adaptive Server Enterprise 11.9.2版本中增加了"追加行"的逻辑，仅对数据加锁和锁存技术的采用，使得索引访问与空间分配得以协调，从而明显提高了性能。现在通过执行这样的任务，就能够不需要再采用依靠数据库加锁的一系列复杂的传统事务就能进行表的追加。

在下列的实验中，基于 TPC/B 的历史记录表被用于确定对堆表追加行时可能提高性能的级别。在每个事务的末尾插入历史记录表，作为工作性能的一条审计记录。运行 TPC/B 的正常过程要采用多张已经分区化的历史记录表。（例如，在第三节中所显示的小表实验中，采用了具有 100 张分区化的 60 张历史记录表)。

这种实验的结果如下图所示：



备注：此处运行号分别对应于在所有直接更新和放置适当位置的小表测试中的运行号。

如果分析加锁争用与冲突，可以观察到排他性的组块加锁大大减少，这是由于对于"下一行加锁"的技术已经被增加到按数据行加锁机制的一部分。但是这种功能还没有充分扩展为包括仅对数据加锁的方式，这就是在这些案例中为什么还会发生某些组块现象的原因。

从这些结果，可以看到需要群聚或分区的堆表不止是一个选择，当采用仅对数据加锁方式时将给用户在明显提高其吞吐量的方面一个很实际的替换方式，特别是在数据行加锁方式的情况下，更是如此。

2) 追加到非群聚性索引的叶级末端

正如在 4.1 节中所描述的，对于堆表页链末端的热点（hot spots）可以通过使用群聚性的索引或分区来加以解决。但是当非群聚性索引最末级（即叶级）的最后一个页面发生争用冲突时，或者在到达一个特定的点的中途，利用上述技术中的任一种方法都不能使上述问题按传统的方式加以解决。既然按照升序值建立索引是非常普通的（例如序列号、日期/时间列等），这就成为迄今为止难于克服的一个问题.

在采用仅对数据加锁的情况下，索引页不再按照传统方式进行加锁，而宁可使用锁存技术。因为锁存仅仅是一个短时期的同步技术，所以使用传统的页加锁方式时，不会加大开销或可能发生当主要的组块现象。

采用 4.1 节中的历史记录表，设计了一个实验，其中按照日期时间列建立了非群聚性的索引，这个日期时间列的值是在该列插入的时刻使用 getdate() 所得到的。这个按常数升序所建立的索引引起了索引叶级的最后页面的争用冲突。这个表自身被划分为 1000 个分区，以试着避免在任何一个数据页发生冲突。

实验结果如下图所示:



因为在这种情况下前面所提及的议题还是有争论的，也没有考虑加锁的尺度，所以选择仅按数据行加锁或数据页加锁是没有什么实质意义的。正如所看到的，一旦完全通过锁存技术避免对索引加锁的话，那么明显可以使性能得到提高和改善。

3.10.4 死锁

1) 发生死锁的原因

以前，由于采用通过服务器访问或变更数据的技术，就存在可能发生死锁的情况。对于常常是更多地由 ASE 的处理特性所引起的死锁， 而不是由不恰当的事务设计所引起的死锁，就给数据库管理员（DBA）带来了特别的麻烦。通过几种增强方式，ASE 11.9.2 将这些系统所引起的死锁问题得到了解决。

随着锁存技术的引入，对于其它用户而言，索引页只是在很短的周期内不能被使用，永远不会延续到数据库事务中。所以锁存技术的特性将那些在索引页中发生死锁的情形都排除掉了。这一点对在使用仅仅是数据表时，可以使用并发技术的情况具有巨大的效果。

此外，为了进一步处理针对数据可能发生的死锁，已经认识到可以对释放共享锁的时间点加以改变，这样一来，就能够使对以前发生死锁的数据行的并发访问保持一致。在处理一个连接（join）时，以前的处理方式是将在外部表中加锁的行一直保持到

检索到内部行时为止。然而在大多数情况下，一旦检索到数据就不再需要对进行访问的行继续加锁。随着引入固定的行标识（RIDs），这就意味着，外部表的扫描从这一行扫描到下一行，一旦所需要的信息已经被检索到之后，在所连接的外部表中，对行的加锁就可以释放了。这个功能被称之为“带有加锁方式的读提交”（"read committed with lock”），并可用如下的命令加以配置：

```
sp_configure ‘read committed with lock’,0|1
```

2) 客户情形

为了证实上述情形，建立了一个实际的客户环境，以便按照一系列它们所提供的测试进行，用于验证他们所认为的最坏的并发情况下以确保不发生死锁。

在这类“最坏情况”测试中的应用负载如下：

- 在一台应用服务器上执行项分开的流式任务，它们包括如下任务：
  - 大量的插入
  - 在一个单一逻辑事务中进行 5000条 SQL 语句的更新操
- SQR 报告
- 用户执行 OLTP 工作（插入、更新和联机查询）
- 使用复杂查询的报告。向下钻取的报告：
  - 访问一定数量的表（典型为 6 或 7）
  - 引用至少包含两个子查询的 SQL 语句；
  - 在光标定义中采用截然不同的光标（cursors）,以引起工作表的使用。

3) 可能发生的死锁

采用较早版本的服务器产品，进行上述的应用实验，可能发生死锁，其主要原因如下：

插入与非群聚性的索引访问

如果插入一行，服务器就会采用群聚性的索引以访问存贮该行所在的页面，并在一种排它性的页面加锁方式下执行插入操作。接着服务器就对表的每个非群聚性索引进行访问，在一种排它性的页面级加锁方式下，在其末端（叶级）结构中插入相应的条目。如果第二个用户也通过非群聚性的索引访问该表，他们可以按照非群聚性索引保持对该叶级页面实施共享加锁，与此同时，这些非群聚性的索引也由这个排它性的页面加锁所阻碍并用在插入操作中。这种对由第二个用户对非群聚性索引的叶级结构所进行的加锁方式，可能阻碍插入尝试并为第一个用户增加一个条目，从而可能引起死锁。

更新的范围

在这个商业应用的设计中，开始首先在一张表中采用零值与一个标识符两部分组成的键值插入所有的新行。在一个单一的逻辑事务中对所有的行使用相同的标识符来进行并发更新。这些更新采用唯一键值去代替零值，就引发了索引行移动到索引的叶级结构的最后。当多个用户并发执行这个任务时，第一个用户就锁定进行处理的页面和索引所在的叶级结构的末端。这将阻碍用户试图对第二个按行范围所进行的工作。当第一个用户到达所确定的页面边缘，即该范围即将终止，下一批所组块并妨碍的页面范围即将开始之时，就可能发生死锁。

按照不同顺序访问表

即使访问行是不同的，对表按照任何不一致的顺序进行访问，也会意味着可能发生死锁。这是因为一个用户所访问的行可能在另一个用户访问的同一个以不同顺序排列的页面中。对于应用系统而言，当然这种设计明显是很差的，但是却还令人惊奇地普遍存在。

连接与更新

正如在 5.1 节所叙述的有关“采用加锁机制的读提交”段的内容，当处理连接，同时对外部表的加锁方式会一直保持到内部行被检索到为止，这种情况就会发生。如果第二个用户在访问所连接的外部表中的行之前，先访问了所连接的内部表的行，这种操作可能受阻，因为加锁的顺序被连接所破坏，从而也会引起死锁。

4) 问题的解决方案

采用 Adaptive Server Enterprise 11.9.2的产品, 应用对索引页的锁存技术以代替传统的加锁方式，以及更小尺度的按行加锁方式，导致了前三种死锁类型不再发生。采用“按照加锁进行读提交”可以处理第四种死锁情况。

作为进一步的测试，在前面所述及的最坏情况实验中，同时还进行了增加额外负载的运行实验。这类实验包括一个拥有 7000条 SQL 语句的单一事务，和从 Microsoft Excel电子数据表中卸载数据（加入 ASE 服务器）。这两个版本的测试已经进行了多次，从未发生过死锁。

5) 一类新的死锁以及如何解决

就在 Adaptive Server Enterprise 产品的设计中进行修正以防止死锁和增加并发性的同时，也发生了一种情形，可能发生死锁，而这种情况在过去是不可能发生的。

当一次更新操作必须访问整个表时（例如此时不支持索引），可能会发生一种改变。在它们已经被确认为符合要求之后，将只对个别行加锁（以前是对一张表加锁）。当访问小表的时候，对于因为性能的原因而建立的任何索引，一般而言都不会很多。如果一个用户扫描这些小表之一，并对一行加以限定，这将随之带来一个排他性的行加锁。如果第二个用户也访问同一张表，这种扫描就会因为第一个用户的排他性行加锁所阻碍。如果第一个用户试图更新同一张表中的另一行，这种扫描也会被第二个用户所阻碍，所以引起死锁。

为了解决这一问题，已经采用了一种称之为“并发优化”的优化程序。这个方法指明了如果在一张表中需要进行存取行的页面数小于某个固定的数字（其缺省值为 16），那么将不再考虑使用任何一个索引，而宁可对表进行直接扫描。这就意味着对

于很小的表，利用索引而不是用全表扫描方式直接访问某些可能会产生直接影响的行，存在引起死锁的可能性。在考虑扫描一张表之前，需要确定在一张表中能有多少页，可采用如下的命令在页级结构中加以配置：

sp_chgattribute <tablename>, “ concurrency_opt_threshold” , <value>

为了提高性能，也需要仅仅对表中的少数行添加索引的建立。与以前一般都忽略了索引不同，现在利用索引是为了提供最大的并发性并避免死锁的发生。

3.10.5 何时使用不同类型的加锁方法

为了在尽量降低资源消耗的前提下实现性能和并发性的平衡， Sybase 已经建立了集中加锁方式，它们分别具有略有不同的特性。在什么时候采用那一种加锁方式，对这一问题。可以利用下列的“大拇指原则”，基于环境和应用程序的基本情况就应该采用的适当的加锁框架，提出原则性的指导意见。

1) 全页加锁方式

- 除非有特别的理由，应尽量使用全页加锁方式，因为它消耗诸如内存和 CPU 之类的资源最少。
- 如果发生争用冲突，并不是空间的问题，而且也可以排除正在执行的工作负载是由于在特定点（例如热点（hot spot））进行插入或延迟更新所引起的，在全页加锁方式下采用填充因子（Fillfactor）或 max_rows_per_page 将能够在不增加维护问题（重组）的条件下对行级加锁提供类似的性能改善。
- 如果一张堆（栈）表中发生了大量的插入操作，而且空间不成为问题的话，采用较大数目分区的全页加锁方式将使性能提高到一定的水平。
- 如果工作负载已经是同 CPU 紧密相关的，那么附加的仅对数据加锁方式所需要的 CPU 实用程序，可能也是使用与 CPU 较少关联的全页加锁方式的一个主要原因。这一点对于已经使用了填充因子来解决争用冲突问题的情况尤其如此。

2）数据行加锁方式

- 对于那些发生争用冲突而且工作负载基本上是由于延迟更新所引起的情况，应该使用数据行加锁方式，而不要再考虑空间问题。
- 一张表占据数据库空间的总量是一个问题，如果表较小而且有争用和冲突的话，应该采用按行加锁方式。
- 在一列需要非群聚性索引而且该列上所有的插入操作都在这个索引的一个确定点上发生时，按行加锁方式提供了既可以不发生索引加锁争用冲突的情况，又具有所需要的索引的能力。

- 对于那些正在使用堆（栈）表，但空间成为问题的情况，仅对数据加锁方法提供了较高的性能。
- 对于死锁的发生，利用仅对数据行加锁是一个可选的解决方案，其原因是或者由于较低的加锁争用冲突，或者由于采用了加锁方式下的读提交。

3）数据页加锁方式

- 对于那些需要仅对数据加锁但加锁机制的数目是一个问题时（例如由于内存），那么采用仅对数据页加锁可能就是一种最优的选择。
- 当由于某个特定的或罕见的原因需要对一张表实现仅对数据加锁时，那么采用数据页加锁比数据行加锁方式占用的资源较少，而且当加锁争用冲突只是限制在索引时，会有相同或更好的性能。

3.10.6 结论

在推出 Adaptive Server Enterprise 11.9.2产品之前，在 Sybase 公司提供了最高性能的关系性数据库服务器产品的同时，也存在着一组特定的加锁问题限制了用户可能获得的性能。这些问题主要对那些不是针对该产品所专门设计的应用系统有所影响，可能需要在工作中逐步靠近 Sybase产品，或者进行代价昂贵的性能改善以使应用系统更符合 Sybase的技术特性。

随着 Adaptive Server Enterprise 11.9.2新版产品的普遍上市，Sybase 公司正在向广大用户提供已被证明能够解决加锁争用冲突问题的解决方案。他们既能使用户在现存的应用系统中同样达到以前所达到的非常高超的性能水平，同时又向用户提供了一系列可以解决所可能遇到的加锁争用冲突问题的解决方案。

在本技术白皮书中所包含的测试结果，和通过论述这些结果的文献已经论证了新型的加锁和锁存技术，可以使用户在不需要任何应用系统作任何改变的条件下，大幅度地提高性能与可伸缩性。

Sybase已经作出了改变，对于先前或者是在有限的方式，或者是具有相当重要影响作用的方式下，使用 Sybase产品的问题，都提供了采用范围广泛的组合方法以利用 Sybase Adaptive Server Enterprise 11.9.2 版本产品，清除以前所遇到的各种障碍。这样一来，就可以使所有服务器的用户，都能达到以前那些专门为 Sybase环境所特意设计的系统中所能体验到的性能水平。

3.11 DBCC

我们知道，在数据库系统的开发和应用中，必须保证数据库的完整性和一致性。当数据库出现了严重错误；当我们怀疑数据库受到破坏（如无法用 drop命令删除数据库或对象，使用某个表时出现“不可靠数据”的信息等）；当用户改变了 Server 的缺省

排序的顺序或改变了字符集而需要检查；当 SA 对系统做定期检查；这些时候，我们都需要使用数据库一致性检查工具（Database Consistenecy Checker，简称 DBCC）。DBCC 是一个实用命令集，用来检查一个数据库的逻辑一致性及物理一致性。在开发和应用中，DBCC 是我们经常要使用的命令。

DBCC 命令的格式如下

```
dbcc
(checktable ((表名|表标识( [, skip_ncindex] ) |
checkdb [(数据库名[, skip_ncindex] )] |
checkalloc [ (数据库名[, fix | nofix] )] |
tablealloc( {表名|表标识}
[, {full |optimized |fast |null}
[, fix |nofix] ]]) |
indexalloc ( {表名|表标识}, 索引标识
[, {full |optimezed | fast | null}
[, fix |nofix ]] ) |
checkcatalog [ (数据库名)] |
dbrepair(数据库名,dropdb ) |
reindex({表名|表标识} ) |
fix_text({表名|表标识} )
```

dbcc 的权限，对于 checktable,fix_text 和 reindex 是缺省赋给表的属主，对于 checkdb,checkalloc，checkcatalog,dbrepair，indexalloc 和 tablealloc，是缺省赋给数据库属主的。DBO 自动获得 DBCC 命令和全部选项的权限。该权限不可转授。此外，dbcc 在数据库是活动时运行，除了 dbrepair 选项和带有 fix 选项的 dbcc checkalloc 以外。

checktable 选项

checktable 是用来对一个指定的表做检查，确保索引和数据页正确地连接，索引按正确的顺序存储，所有指针的一致性，每页上数据信息的合理性，页偏移的合理性。如果日志段在它自己的（日志）设备上，对 syslogs 表使用 dbcc checktable 命令可以报告已使用的和剩余的日志空间，使用 skip_ncindex 选项使得 dbcc checktable 跳过对用户表上非聚簇索引（nonclustered index）的检查。缺省是检查所有的索引。

例１.检查日志使用的空间量和未用的空间量：

dbcc checktable (syslogs)

若日志段在日志设备上，则会返回如下信息：

```
checking syslogs
The total number of data page in the table is 1.
```

```
NOTICE:Space used on the log segment is 0.20 Mbytes, 0.13%.
NOTICE:Space free on the log segment is 153.4Mbytes,99.87%.
DBCC execution Completed.If dbcc printed error messages,
Contact a user with SA role.
```

若日志不在它自己的设备上，则会显示下列信息：

```
NOTICE:Notification of log space used/free.
Can not be reported because the log segment is not on its own device.
例 2. dbcc checktable (titles)
The total number of data page in this table is 3.
Table has 18 data rows.
DBCC execution Completed. If DBCC printed error messages. contact a user
with SA role.
```

checkdb 选项

运行 checkdb 选项同 checktable 检查的内容一样，但它是对一指定数据库中的每张表都做这样的检查。若未指定数据库名，checkdb 检查当前的数据库。checkdb 返回的信息，也同于 checktable。

checkalloc 选项

checkalloc 是检查指定数据库，看其所有正确分配的页和尚未分配的页的情况。若未指定数据库名，则 checkalloc 检查当前数据库。checkalloc 会返回已分配的和使用的空间数量。checkalloc 的缺省模式为 nofix，要使用 fix 选项，必须把数据库置于单用户模式。 例：

```
dbcc checkalloc (pubs2)
.
.
.
alloc page 0 (#of extent=32 used pages=68 ref pages=68)
alloc page 256 (# of extent=32 used pages=154 ref pages=154)
alloc page 512 (# of extent=28 used pages=184 ref pages=184)
alloc page 768 (# of extent=1 used pages=1 ref pages=1)
total (# of extent=93 used pages=407 ref pages=407) in this database.
DBCC execution completed.If dbcc printed error message,
Contact a user with System Adminstrator (SA) role.
```

tablealloc 选项

tablealloc 检查指定的表以确保所有页都被正确地分配。它是 checkalloc 的缩小版本。对单张表进行相同的完整性检查。使用 tablealloc 可以生成三种类型的报表：full,optimized 和 fast。full 选项相当于表一级的 checkalloc；它报告各种类型的分配错误。optimized 选项基于表的对象分配映像（OAM）页里列出的分配页生成报告。它并不报告，也不能整理 OAM 页里没有列出的在分配页上没有引用的扩展（extent）。如果没有指明类型，或使用了 null，则 optimized 选项是缺省的设置。fast 选项，并不生成分配报告，但生成一个被引用但并没有在扩展里分配的页的额外的报告。fix|nofix 选项决定 tablealloc 是否整理表中发现的分配错误。对于所有的表，缺省为 fix，但系统表除外，它们的缺省为 nofix。要对系统表使用 fix 选项，必须首先将数据库置成单用户模式。 例：

```
dbcc tablealloc(titles)
显示信息如下：
The default report option of OPTIMIZED is used for this run. The default
fix option of FIX.is used for this run.
.
.
.
Total #of extent=3
Alloc page 256 (# of extent=1 used pages=2 ref pages=2).
Alloc page 256(# of extent=1 used pages=2 ref pages=2)
Alloc page 256 (# of extent=1 used pages=2 ref pages=2)
Total (# of extent=3 used pages=8 ref pages=8) in this database.
```

indexalloc 选项

indexalloc 检查指定的索引，确保所有的页都被正确地分配，它是 checkalloc 的缩小版本，对单独一条索引指定同样的完整性检查。其中各选项与 tablealloc 相同。

checkcatalog 选项

checkcatalog 选项用于检查系统表内，系统表之间的一致性。例如：它确保在 syscolumns 表中的每一（数据）类型在 systypes 表中都有一个相匹配的记录；对于 sysobjects 中的每个表和视图在 syscolumns 表中应有关于它们每一列的描述记录；确保在 syslogs 中的最后一个检查点是有效的。checkcatalog 也报告任何已定义的段。若不指定数据库名，则检查当前数据库。

dbrepair 选项

dbrepair（数据库名，dropdb）选项是删除一个受破坏的数据库。受破坏的数据库是不能用 drop database 命令删除的，drop database 只能删除正常的数据库，当执行 dbrepair 命令时，任何用户（包括执行此命令的用户）都不得使用正被删除的数据库。该选项要在 master 库中运行。

reindex选项

reindex 选项通过运行 dbcc checktable 的“fast”执行方式检查用户表上索引的完整性。如果它检测出索引有问题则会删除并重建索引。在 ASE 的排列顺序改变之后，SA 或表属主应该执行这一选项。此选项不能在用户定义的事务中运行。例：

```
dbcc reindex (titles)
返回信息：One or more indexes corrupt.They will be rebuilt.
```

fix_text选项

ASE 的字符集由单字节转变为多字节后，fix_text 选项用于升级文本值。ASE 的字符集由单字节转变为多字节字符集会使文本数据的管理更加复杂。由于文本值可能较大足以覆盖若干页，ASE 必须能处理（通过页约束）可能横跨页的字符。为做到这点，服务器需要在每一文本页上添加一些信息。SA 或表属主必须在文本数据的每一个表上运行 dbcc fix_text，以计算所需要的新页数。

总之，DBCC 命令所返回的信息能准确地反映数据库及它的各个对象的状态，是我们检测数据库的好帮手。

3.12 Sybase ASE 的字符集

3.12.1 概念

一、什么是字符集？

字符集是字符（包含字母，数字，符号和非打印字符等）以及所指定的内码所组成的特定的集合。通常一个字符集包含一个字母表中的字符，例如拉丁字母表被使用在英语语言中，那么如果要使用拉丁字符，就要配置使用英语语言集中的特定的字符集合——拉丁语言字符集。这里为什么特指了是英语语言集呢？因为字符集是基于某种操作系统平台和某种语言集支持的。语言集的集合被称为语言组，它可能包含一种或多种语言。本地字符集是基于特定语言组中所包含的一种或多种语言支持的，在特定操作系统平台上编码的集合。

在 Client/Server 系统中，支持多语言的数据处理，但是所有的语言必须属于同一个语言组。例如，从下表可以看出，如果服务器中的数据用组 1 中的字符集，则同一数据库中可以有法语，德语，英语以及该组中的其它语言。而在这个数据库中就不能同时存储日语，法语了。

这里请注意一个非常特别的字符集—Unicode—它支持世界上超过650种语言的国际字符集。Unicode允许在同一服务器上混合使用不同语言组的不同语言。

表1－1　Adaptive Server支持的语言和字符集

语言组	语言	字符集
组 1	*西欧：* 阿尔巴尼亚语、加泰罗尼亚语、丹麦语、荷兰语、英语、法罗语、芬兰语、法语、加利西亚语、德语、冰岛语、爱尔兰语、意大利语、挪威语、葡萄牙语、西班牙语、瑞典语	ASCII 8, CP 437, CP 850, CP 860, CP 863, CP 1252[a] , ISO 8859-1, ISO 8859-15, Macintosh Roman, ROMAN8
组 2	*东欧：* 克罗地亚语、捷克语、爱沙尼亚语、匈牙利语、拉脱维亚语、立陶宛语、波兰语、罗马尼亚语、斯洛伐克语、斯洛语尼亚语（和英语）	CP 852, CP 1250, ISO 8859-2, Macintosh Central European
组 4	波罗的语（和英语）	CP 1257
组 5	*古斯拉夫语：* 保加利亚语、白俄罗斯语、马其顿语、俄语、塞尔维亚语、乌克兰语（和英语）	CP 855, CP 866, CP 1251, ISO 8859-5, Koi8, Macintosh Cyrillic
组 6	阿拉伯语（和英语）	CP 864, CP 1256, ISO 8859-6
组 7	希腊语（和英语）	CP 869, CP 1253, GREEK8, ISO 8859-7, Macintosh Greek
组 8	希伯来语（和英语）	CP 1255, ISO 8859-8
组 9	土耳其语（和英语）	CP 857, CP 1254, ISO 8859-9, Macintosh Turkish, TURKISH8
组 101	日语（和英语）	CP 932 DEC Kanji, EUC-JIS, Shift-JIS
组 102	简体中文（PRC）（和英语）	CP 936, EUC-GB
组 103	繁体中文（ROC）（和英语）	Big 5, CP 950[b] , EUC-CNS
组 104	韩语（和英语）	EUC-KSC
组 105	泰语（和英语）	CP 874, TIS 620
组 106	越南语（和英语）	CP 1258
Unicode	超过 650 种语言	UTF-8

a. 除了 0x80-x9F 代码点在 CP 1252 中映射为字符以外， CP 1252 与 ISO 8859-1 是相同的。
b. CP 950 与 Big 5 相同。

注意：表中所显示的所有字符集，因为任何字符集的前128（十进制）个字符都包含拉丁字母表，所以所有字符集都支持英语。各字符集中前128个字符之外的字符各不相同，用于支持不同的本地语言字符。

二、什么是排序顺序？

每种字符集都有一种或多种排序顺序，Adaptive Server 使用它们存储数据。排序顺序与特定的语言或语言组及特定的字符集联系密切，不同的语言对同样字符的排序是不同的，因此，需要特定语言的排序顺序，以便正确地对字符进行排序。另外，排序顺序与特定地字符集也密切相关，对于特定字符集可使用的排序顺序位于字符集目录的排序顺序定义的文件中(.srt 文件)。

有关字符集及其可用排序顺序的列表，如下所示：

表 1－2 可用的排序顺序

语言或脚本	字符集	排序顺序
所有语言	UTF-8	二进制
古斯拉夫语： 保加利亚语、白俄罗斯语、马其顿语、俄语、塞尔维亚语、乌克兰语	CP 855, CP 866, CP 1251, ISO 8859-5, Koi8, Macintosh Cyrillic	字典顺序，区分大小写，区分重音
英语、法语、德语	ASCII 8, CP 437, CP850, CP 860, CP 863, CP 1252a, ISO 8859-7, ISO 8859-5, Macintosh Roman, ROMAN8	字典顺序，区分大小写，区分重音 字典顺序，不区分大小写，区分重音 字典顺序，区分大小写，区分重音，具有优先级 字典顺序，不区分大小写，不区分重音
英语、法语、德语	CP 850	替代字典顺序，区分大小写 替代字典顺序，区分大小写，不区分重音 替代字典顺序，区分大小写，具有优先级。
希腊语	ISO 8859-7	字典顺序，区分大小写，区分重音
匈牙利语	ISO 8859-2	字典顺序，区分大小写，区分重音 字典顺序，不区分大小写，区分重音 字典顺序，不区分大小写，不区分重音
俄语	CP 855, CP 866, CP 1251, ISO 8859-5, Koi8, Macintosh Cyrillic	字典顺序，区分大小写，区分重音 字典顺序，不区分大小写，区分重音
斯堪的纳维亚语	CP 850	字典顺序，区分大小写，区分重音 字典顺序，不区分大小写，具有优先级
西班牙语	ASCII 8, CP 437, CP850, CP 860, CP 863, CP 1252, ISO 8859-1, ISO 8859-15, Macintosh Roman, ROMAN8	字典顺序，区分大小写，区分重音 字典顺序，不区分大小写，区分重音 字典顺序，不区分大小写，不区分重音
泰语	CP 874, TIS 620	字典顺序
土耳其语	ISO 8859-9	字典顺序，区分大小写，区分重音 字典顺序，不区分大小写，不区分重音 字典顺序，不区分大小写，区分重音

排序顺序用于

——创建索引

——将数据存入按索引排序的表

——指定 order by 子句

对于不同类型排序顺序的解释

二进制排序顺序：

对于所有字符集都至少提供一个二进制排序顺序，这一排序顺序基于字符集中分配给代表每个字符的代码（“二进制”代码）的算数值，适用于每个字符集的前 128 个字符和亚洲语言。当字符集支持一种以上的语言时，二进制排序顺序将会得出不正确的结果，这时就应该选择其它排序顺序了。

字典排序，区分大小写，区分重音：

分别对大写和小写字母进行排序。字典排序顺序识别字母的各种重音形式，并将它们排在相关联的非重音字母之后。

字典排序，不区分大小写，区分重音：

按字典顺序排序，大写字母与小写字母等同，在排序结果中大小写字母混合使用。对于避免表中名称的重复条目很有用。

字典排序，不区分大小写，区分重音，具有优先级：

在排序时不区分大小写，在所有其它条件相同时，大写字母具有高的优先级（即大写字母先出现）。

当 order by 子句中指定的列与表的聚簇索引键值相匹配时，使用这种排序顺序可能导致大表性能降低，因此，不建议使用这种排序顺序，除非特意要求大写字母排在小写字母之前。

字典排序，不区分大小写，不区分重音：

将加重音格的字母与未加重音格的相关联字母同等对待，它在排序中混合了重音字符。

三、什么是字符集转换？

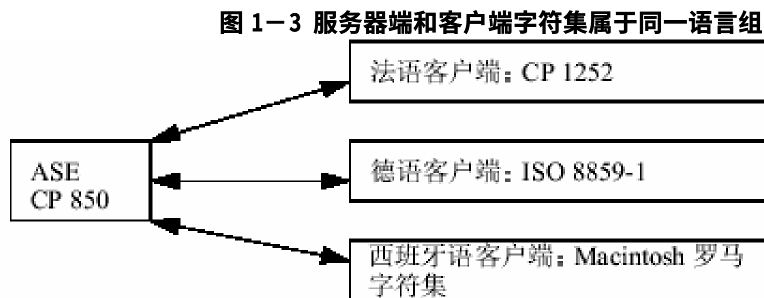
为保持客户端与服务器之间的数据完整性，数据必须在字符集之间进行转换，目的是跨机器和字符集使用时，确保“a”还是“a”，此过程就是字符集转换。

<u>字符集转换的方式</u>

本地字符集的转换：

Adaptive Server 支持属于同一语言组的本地字符集之间的转换。如果服务器把一种本地字符集作为它的缺省值，则客户端字符集必须属于同一个语言组，此时可以在服务器上浏览所有客户端提交的数据。如下图：

图 1－3 服务器端和客户端字符集属于同一语言组

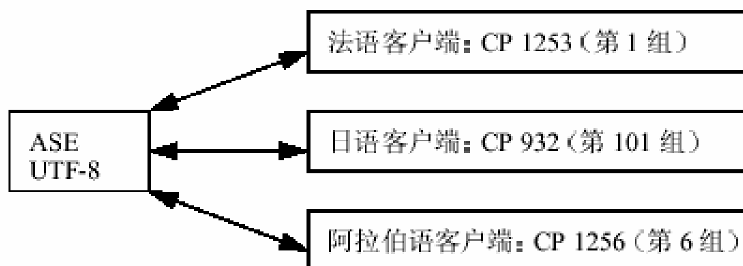


本图中，服务器端与客户端使用的语言集与字符集都同属于组 1(见表 1－1)，那么他们之间实现的就是本地字符集转换方式。

Unicode 系统中的转换方式：

在 Unicode 系统中，由于服务器的缺省字符集为 UTF －8，所以客户端字符集可以是任何语言组中的一种本地字符集。如下图：

图 1－4Unicode系统中的字符集转换



上图中来自每一个客户端的数据经过服务器和每个客户端时，都会被正确转换，而无论每一个客户端选择的时哪个语言组的字符集，原因是 ASE 服务器端选择了缺省字符集 UTF-8.

字符集转换类型

直接转换：

支持同一语言组内两种本地字符集之间的转换。例如，Adaptive Server 支持 CP437 与 CP850 之间的转换，因为他们同属于第 1 语言组。

Unicode 转换：

Unicode 转换可应用于所有本地字符集，在两种本地字符集之间进行转换时，Unicode 转换方式把 Unicode 作为中间字符集。例如，在服务器缺省字符集 CP437 和客户端字符集 CP860 之间进行转换时，CP437 先被转换成 Unicode，Unicode 再转换成 CP860。

Unicode 转换方式既可以用于服务器缺省字符集 UTF －8，还可用于本地字符集。除非使用服务器缺省字符集 UTF －8，否则您必须专门配置您的服务器才能使用 Unicode 转换方式（配置方法请参看配置篇之如何配置字符集的转换类型）。

如何选择字符集的转换方式

这将取决于系统的类型。

在非 Unicode 系统中，服务器和客户端的字符集为本地字符集，因此可以使用 Adaptive Server 直接转换，但是有些字符集没有直接转换，这种情况就必须使用 Unicode 转换了。

如下表所示：

表 1－5 字符集转换方式

	列 1	列 2
语言组	Adaptive Server 直接转换和 Unicode 转换	仅限 Unicode 转换
组 1	CP 437、CP 850、ISO 8859-1、Macintosh Roman、ROMAN8	CP 860、CP 1252、ISO 8859-15、CP 863
第 2 组	CP 852、CP 1250、CP 8859-1、Macintosh Central European	ISO 8859-2
第 4 组	无需转换（仅支持一个字符集）	
第 5 组	CP 855、CP 866、CP 1251、ISO 8859-5、Koi8、Macintosh Cyrillic	
第 6 组		CP 864, CP 1256, ISO 8859-6
第 7 组	CP 869, CP 1253, GREEK8, ISO 8859-7, Macintosh Greek	
第 8 组		CP 1255, ISO 8859-8
第 9 组	CP 857, CP 1254, ISO 8859-9, Macintosh Turkish, TURKISH8	
第 101 组	DEC Kanjii、EUC-JIS、Shift-JIS	CP 932
第 102 组		CP 936、EUC-GB
第 103 组		Big 5、CP 950、EUC-CNS
第 104 组	无需转换（仅支持一个字符集）	
第 105 组		CP 874, TIS 620
第 106 组	无需转换（仅支持一个字符集）	
Unicode	无需转换（仅支持一个字符集）	

——如果系统中使用的所有字符集都在表 1－5 的列 1 中，则使用直接转换。前提是所有字符集同属于一个语言组。

——如果系统中使用的所有字符集都在表 1－5 的列 2 中，或者有些在列 1 中，有些在列 2 中，那么必须配置服务器使用 Unicode 转换方式。前提是所有字符集同属于一个语言组。

在 Unicode 系统中，如果服务器缺省字符集为 Unicode UTF－8，那么所有的转换将在 UTF－8 与客户端使用的本地字符集之间进行，因此，在 Unicode 系统中，只能使用 Unicode 转换。

3.12.2 配置

一、 如何配置字符集的转换类型

<u>禁用字符集转换的配置方法</u>

在 isql 环境中执行：

```
1>   sp_configure “disable character set conversion”,1
```

```
2>   go
```

“disable character set conversion”参数默认配置值为 0，即启用字符集转换。

<u>如何配置字符集的转换类型</u>

把”enable unicode conversions”参数设置为 1 或者 2。当配置为 1 时，此设置使用直接转换或 Unicode 转换；当配置为 2 时，此设置使用 Unicode 转换；默认配置值为 0，使用直接转换。

在 isql 环境中执行：

```
1>   sp_configure ”enable unicode conversions”,1
2>   go
```

二、如何配置服务器端缺省字符集

<u>直接转换法</u>

直接转换法是指直接使用 Sybase 提供的实用程序，如 UNIX 平台上使用 sqlloc 命令或者编辑 sqlloc.rs 脚本文件；Windows 平台使用“服务器配置”图形化管理工具直接配置服务器端字符集。

使用直接转换法的条件是：
——服务器中没有用户数据
——对服务器中用户数据的损坏是可以接受的
——绝对确定服务器中的数据只使用 ASCII-7 字符集

<u>间接转换法</u>

间接转换法相对于直接转换法而言，需要通过一下三步来完成配置工作：
1．先将服务器端的数据使用 bcp 命令导出
2．再选择直接转换法之中的一种方式配置服务器端字符集
3．再使用带有－J 参数的 bcp 命令把数据导回服务器端

<u>配置服务器端字符集的方法</u>

sqlloc—适用于 UNIX 平台的命令

在$SYBASE_OCS/bin 目录下执行：sqlloc，将出现一个图形化界面，在这个界面中您可以直接选择语言集，字符集，排序顺序，便很容易地就完成了配置工作。

编辑 sqlloc.rs 脚本文件

将$SYBASE-ASE/init/sample_resource_files/sqlloc.rs 拷贝到$SYBASE_OCS/bin 目录下，按以下黑体字提示编辑该文件：

sybinit.release_directory: /home/sybase———输入 Sybase 产品的安装路径
sqlsrv.server_name: SYB125———输入数据库服务器的名称
sqlsrv.sa_login: sa
sqlsrv.sa_password: ———输入 sa 的口令，若为空，则什么也不填
sqlsrv.default_language: us_english———输入想要配置的语言集
sqlsrv.language_install_list: USE_DEFAULT

```
sqlsrv.language_remove_list: USE_DEFAULT
sqlsrv.default_characterset: cp850———输入想要配置的字符集
sqlsrv.characterset_install_list: USE_DEFAULT
sqlsrv.characterset_remove_list: USE_DEFAULT
sqlsrv.sort_order: binary———输入想要配置的排序顺序

# An example sqlloc resource file...
# sybinit.release_directory: USE_DEFAULT
# sqlsrv.server_name: PUT_YOUR_SERVER_NAME_HERE
# sqlsrv.sa_login: sa
# sqlsrv.sa_password:
# sqlsrv.default_language: french
# sqlsrv.language_install_list: spanish,german
# sqlsrv.language_remove_list: USE_DEFAULT
# sqlsrv.default_characterset: cp437
# sqlsrv.characterset_install_list: mac,cp850
# sqlsrv.characterset_remove_list: USE_DEFAULT
# sqlsrv.sort_order: dictionary
```

保存已经修改好的 sqlloc.rs 脚本文件，执行以下命令：

```
sqllocres -r sqlloc.rs
```

注意屏幕上出现的提示信息，如无异常，则完成配置工作。

“服务器配置”图形化管理工具——适用于 Windows 平台

“服务器配置”管理工具提供了一个易于操作的图形化管理平台，根据工具中提示的信息，很容易就完成了字符集的配置工作，这里就不多讲了，请参看相关文档说明。

三、如何配置客户端缺省字符集

配置客户端缺省字符集实际上就是对“$SYBASE\locales”目录下 locales.dat 文件的修改。

在 Windows 平台用写字板方式打开该文件，在 UNIX 平台可以直接使用“vi”命令打开该文件，我们会看到，该文将中所有字符集的配置都是以服务器端操作系统平台名称分组的：

```
.
.
[aix]
      locale = C, us_english, iso_1
      locale = En_US, us_english, iso_1
      locale = en_US, us_english, iso_1
      locale = default, us_english, iso_1
```

```
        locale = En_US.IBM-850, us_english, cp850
        locale = en_JP, us_english, eucjis
        locale = Fr_FR, french, cp850
        .
        .
[axposf]
        locale = C, us_english, iso_1
        ; Use Posix Locales, straight from the Posix Guidelines
        locale = en_US.88591, us_english, iso_1
        locale = fr_FR, french, iso_1
        locale = zh_CN, chinese, eucgb
        locale = zh_TW, tchinese, euccns
        locale = ko_KR, korean, eucksc
        locale = us_english.utf8, us_english, utf8
        locale = default, us_english, iso_1
        .
        .
```

其中，操作系统名称放在每一组最开始的“[]”中，而且请注意上面黑体字，每一组中都会存在一行“locale = default,…”。我们要修改客户端的默认字符集，就是对这一行进行修改。

例如，某系统服务器端是 SUN 平台，服务器端语言集为 english，字符集为 cp850。我们要修改客户端字符集与服务器端一致，怎么做？

首先找到[SUN]操作系统分组，然后修改“locale = default,…”为“locale = default,us_English,cp850”。

修改前：

```
[sun]
        ; from JLE, KLE, CLE, OS/4.1.1, man setlocale()
        ; and Sun Software Internationalization Guide (p/n 800-5972-08)
        ; use setenv LC_CTYPE, LC_MESSAGES, LANG
        locale = C, us_english, iso_1
        locale = fr, french, iso_1
        locale = de, german, iso_1
        locale = tr, us_english, iso88599
        locale = zh, chinese, eucgb
        locale = zh_CN, chinese, eucgb
        locale = zh_TW, tchinese, euccns
        locale = ko, korean, eucksc
        locale = us_english.utf8, us_english, utf8
        locale = default, us_english, iso_1
```

修改后：

```
[sun]
        ; from JLE, KLE, CLE, OS/4.1.1, man setlocale()
```

```
; and Sun Software Internationalization Guide (p/n 800-5972-08)
; use setenv LC_CTYPE, LC_MESSAGES, LANG
locale = C, us_english, iso_1
locale = fr, french, iso_1
locale = de, german, iso_1
locale = tr, us_english, iso88599
locale = zh, chinese, eucgb
locale = zh_CN, chinese, eucgb
locale = zh_TW, tchinese, euccns
locale = ko, korean, eucksc
locale = us_english.utf8, us_english, utf8
locale = default, us_english, cp850
```

保存该文件，就完成对客户端字符集的修改了。

这里，还要说明一种特殊情况：

为了满足服务器端某些应用的特殊需求，在服务器端设置了一个环境变量：LANG，此时客户端字符集该如何设置呢？

例如，某系统服务器端是 Windows 平台，使用语言集 english，字符集 iso_1，并设置环境变量 LANG=C。我们要修改客户端字符集与服务器端一致，怎么做？

首先找到[NT]操作系统分组，然后在该组中加入一行

“locale = C,us_English,iso_1”

修改前：

```
[NT]
    locale = enu, us_english, iso_1
    locale = fra, french, iso_1
    locale = deu, german, iso_1
    locale = japanese, japanese, sjis
    locale = chs, chinese, eucgb
    locale = cht, tchinese, big5
    ; locale = kor, korean, eucksc
    locale = us_english.utf8, us_english, utf8
    locale = default, us_english, iso_1
```

修改后：

```
[NT]
    locale = enu, us_english, iso_1
    locale = fra, french, iso_1
    locale = deu, german, iso_1
    locale = japanese, japanese, sjis
    locale = chs, chinese, eucgb
    locale = cht, tchinese, big5
    ; locale = kor, korean, eucksc
    locale = us_english.utf8, us_english, utf8
    locale = default, us_english, iso_1
```

```
locale = C,us_english,iso_1
```

因此在修改客户端字符集之前，请先查看服务器端是否设置了环境变量“LANG“，再决定如何修改。

四、如何选择 ASE 字符集使之支持中文字符

目前在 ASE 12.5 中支持中文字符的字符集有四种：CP936，EUCGB，UTF-8 和 GB18030。

其中 EUCGB 字符集是基于 GB2312-80 编码规范的，它的 EUC (Extended Unix Code) 编码范围是第一字节 0xA1~0xFE(实际只用到 0xF7)，第二字节 0xA1~0xFE。

CP936 字符集是基于 GBK 编码规范（实际上的国家标准是 GB13000-90），是对 GB2312 进行的扩展，第一字节为 0x81~0xFE，第二字节分两部分，一是 0x40~0x7E，二是 0x80~0xFE。其中和 GB2312 相 同的区域，字完全相同。

GB18030字符集（国家标准号是 GB18030-2000）是 2000 年 3 月 17 日发布的新的中文编码标准。它是 GB2312 的扩充，采用单/双/四字节编码体系结构，收录了 27000 多个汉字以及藏文、蒙文、维吾尔文等主要的少数民族文字。Sybase 从 ASE 12.5.0.3 之后开始支持 GB18030 字符集。

UTF-8字符集是现有 ASCII系统向 Unicode 转换的一个过渡方案。它使用 1-3 字节表示一个字符。简体中文的每个字符在 utf8 中的长度基本上都是 3 个字节。它的最主要的优点是可以同时支持超过 650 种语言的字符。缺点是针对中文字符来说，需要增加 50% 的空间用来存储。

一般来说，由于 EUCGB 不支持国标一、二级字库以外的汉字，所以我们推荐用户在服务器端和客户端都使用 CP936字符集，或者在 ASE 12.5.0.3 之后还可以使用 GB18030 字符集，它可以支持一些比较生僻的汉字。它的不足是只有一种排序方式，即区分大小写的 Binary方式。所以，如果需要使用支持中文字符集且不区分大小写的数据库时，就只能使用 UTF-8作为服务器端字符集，而客户端使用 CP936或 GB18030 字符集。

另外，还有一种选择是，服务器端和客户端都使用 iso_1 字符集，虽然 iso_1 字符集并不直接支持中文字符，但我们将服务器端和客户端都设置成 iso_1 字符集后，系统也不会在客户端和服务器端进行字符转换，它只不过将一个汉字的两个字节当做两个单独字符来处理，一般情况下没有问题。但当执行 like 匹配查询的时候，它有可能返回不正确的结果，原因是服务器端是根据单字节去匹配查询条件的，很可能会有前一个汉字的第二字节与后一个汉字的第一字节的内码组合符合查询条件，被服务器端作为查询结果返回来。

如果客户端是通过 JDBC 访问数据库的，那么，为了支持中文字符，服务器端最好采用 CP936/GB18030 或 UTF-8作为字符集。如果应用是多层结构的，那么应用服务器的字符集也最好采用与数据库服务器端相同的字符集。

五、如何查看服务器端、客户端字符集

查看服务器端字符集：
在 isql 环境中执行：

```
1>  sp_helpsort
```

```
2>    go
```

查看客户端字符集：
在 isql 环境中执行：

```
1>    select @@client_csname
2>    go
```

3.12.3 错误处理

为什么会出现字符集转换失败？

1．当字符存在于客户端字符集中但在服务器字符集中不存在时，Adaptive Server 的字符集转换将报告转换错误，反之亦然。

用户会碰到下面的错误消息：

Msg 2402,Severity 16 (EX_USER):

Error converting client characters into server's character set. Some character(s) could not be converted.

转换错误会阻止插入与更新语句的执行。如果发生此情况，请检查数据中有问题的字符并替换它们。

2．当客户端发送数据时 Adaptive Server 遇到转换错误，它用 ASCII 码的问号（？）代替可疑字符所占字节，但查询批处理继续进行直到完成为止。

语句完成后，Adaptive Server 将发送一下消息：

Msg 2403,Severity 16 (EX_USER):

WARNING! Some character(s) could not be converted into client's character set. Unconverted bytes were changed to question marks (`?')。

3．当在客户端查询服务器端存储的数据时，当碰到中文汉字，在客户端显示乱码的现象，且没有任何提示信息。

这是我们在应用中经常会碰到的现象，产生的原因是：客户端与服务器端字符集不符。怎么解释呢？假设我们先期设置服务器端字符集为 iso_1，客户端字符集也为 iso_1,然后我们从客户端向服务器端录入了所有的数据；之后当我们需要查询时，使用的客户端，它的字符集为 cp850，那么势必在该客户端上显示的字符集为乱码。

当出现这种情况时，最好配置客户端字符集为先期客户端使用的字符集或者配置客户端字符集与服务器端字符集一致，使得客户端字符集与服务器端字符集匹配。这里我们不建议用户修改服务器端字符集，因为服务器中此时已经存在大量的数据，在使用直接转换方式时，由于源字符集与目的字符集中可能存在无法转换的字符而导致 Adaptive Server 无法启动；如果使用间接的转换方式，会增加工作量。

3.12.4 附：如何安装cp936字符集

以在 Windows 平台安装 cp936 字符集为例，说明如何安装使用服务器中没有被默认安装的字符集。（在 ASE 12.5.0.3 版本中安装 GB18030 字符集的方法类似）

(这里 SYBASE 的安装路径为 c:\sybase)

1.c:\>cd \sybase\charsets\cp936

2.c:\sybase\charsets\cp936> charset -Usa -Psa_pass -Sserver_name binary.srt cp936

3.在 SQL 环境中

1>select name,id from syscharsets

2>go

找到 name 为 cp936 对应的 id(假设为 171)

4.1>sp_configure "default character set id",171

2>go

5.重启 server 两次

（注:第一次启动后，server 会自动宕掉，需要第二次重启后才能使用）

4. 提高篇

4.1 为什么数据库事务日志满了，使用 dump tran with no_log 仍不能截断日志？

有两种情况，可能出现这个问题。一是应用系统给 ASE 发送了一个用户自定义事务，一直未提交，这个最早活跃事务阻碍系统截断日志。二是客户端向 ASE 发送了一个修改数量大的事务，清日志时，该事务还正在执行之中，此事务所涉及的日志只能等到事务结束后，才能被截掉。

对于第一种情况，只要督促用户退出应用或者提交事务，系统管理员便可清掉日志。因为给 ASE 发送 Dump transaction with no-log 或者 with truncate-only，它截掉事务日志的非活跃部分。所谓非活跃部分是指服务器检查点之间的所有已提交或回退的事务。而从最早的未提交的事务到最近的日志记录之间的事务日志记录被称为活跃的。从此可以看明，打开的事务能致使日志上涨，因为在最早活跃事务之后的日志不能被截除。

对于第二种情况，道理也同上。只是在处理它时，需慎重从事。如果这个大事务已运行较长时间，应尽量想法扩大数据库日志空间，保证该事务正常结束。若该事务被强行回滚，ASE 需要做大量的处理工作，往往是正向执行时间的几倍，系统恢复时间长，可能会影响正常使用的时间。

4.2 如何后备数据量大于 2GB 的数据库

当在后备数据量大于 2GB 的数据库时，可能会遇到以下错误：

- **I/Oerror:**
- **operating system error,server device /backup/data. code 27 messages**
- **file too large.**

这是由于后备文件的大小超出了操作系统的用户最大文件限制。而有些操作系统不支持大于 2GB 的文件，这时可以使用 Backup Server 将一个数据库后备到多个文件中。

- **dump database pubs2 to "/usr/sybase/pubs2_dump.1"**
- **stripe on "/usr/sybase/pubs2_dump.2"**
- **stripe on "/usr/sybase/pubs2_dump.3"**
- **go**
-

这种方法还可以提高后备及恢复的速度，但注意恢复也必须用相应多的设备。例如：

```
1>load database pubs2 from "/usr/sybase/pubs2_dump.1"
2>stripe on "/usr/sybase/pubs2_dump.2"
3>stripe on "/usr/sybase/pubs2_dump.3"
4>go
```

4.3 如何更改 ASE 名称（在 UNIX、OPENVMS 平台上）

在 SYBASE 产品中没有特定的函数或者存储过程用来更改 ASE Server 的名称，因此，只能手工修改某些参数或者配置来完成此任务。

需要修改 interfaces 文件；更改 RUN_server_name 文件名，并修改其内容，例如：-s（UNIX）、/SERVER(OpenVMS)后面所跟的参数（server 名称）；更改配置文件名；更改 errorlog 文件名（如果需要）；如果 server 名称加在了表 sysservers 中，需要更改'srvname'、'srvnetname'列，可使用 sp_dropserver、sp_addserver 存储过程来实现。

在 OpenVMS 系统中还要修改 RUN_server_name 文件中 DSLISTEN 的逻辑名称，以及在使用 startserver 过程中/SERVER 后面所跟的参数。然后，重启 SQL/ASE Server。确认 DSQUERY、DSLISTEN 环境变量已经更改为所需内容。

另外，如果 SQL/ASE Server 是被设置为远程服务器，还要修改与此 server 相关的系统表 sysservers 中的字段以及 interfaces 文件。

理论上例如配置文件、RUN 脚本、errorlog 文件不需要指定 server 名称（但是通常习惯指定 server 名称）。在 RUN startup 脚本中要指定 server 名称。

您可以根据以下提纲完成此项任务：

- 如果 server 名称加在了表 sysservers 中，用 sp_dropserver 删除。
- Shut down server
- 编辑 interfaces 文件
- 更改 RUN_server_name 文件名，并修改其内容，-s（UNIX）、/SERVER(OpenVMS)后面所跟的参数（server 名称）
- 更改 errorlog 文件名（如果需要）
- 修改 DSQUERY、DSLISTEN 环境变量（如果需要）
- 更改配置文件名（server_name.cfg、server_name.bak、server_name.nnn），在 SYBASE 安装路径下
- Start server
- 使用 sp_addserver 重新添加 server(如果需要)
- 如果 server 是作为远程 server 使用的，还需要删除并重新添加。

4.4 如何在 HP 平台 BCP OUT 超过 2G 的数据

运行环境：

HP UNIX11,
SYBASE ASE12.0.02 SWR 9310
SYBASE OPEN CLIENT 11.1.1 EBF8633
SYBASE OPEN CLIENT 12.0

问题描述：

SYBASE数据库中一个表BCP OUT时，输出文件大小到达2G时出错，不能完成对此表的备份。具体现象表现为两天对同一张大表BCP OUT输出文件的字节数相同，没有完成所有表的BCP OUT。

问题分析与解决办法：

SYBASE OPEN CLIENT 11.1.1 EBF8960之前的版本不支持HP平台上的超过2G的BCP（OUT，IN）。OPEN CLIENT11.1.1 EBF8960 之后的版本，可以通过“管道”的方式支持文件大小在2G以上的BCP（OUT，IN）。

方法如以下所示：

```
/usr/sbin/fsadm -F vxfs -o largefiles /dev/...         //使文件系统支持大于2G的文件
/usr/sbin/mount -F vxfs -o largefiles /dev/...

mkfifo bcppipe                                   // 建立一个管道
bcp tempdb..large_table out bcppipe -Usa -P -c &  // 后台运行 BCP OUT
cat bcppipe > bcp.out                            // 把 BCP OUT 输出从定向到可支持大
                                                 // 文件的文件系统
```

实施步骤：

1. 使文件系统支持大于2G的文件。
2. 由于现行BCP 备份程序有8个进程同时进行BCP OUT，所以需要建立8个管道 bcppipe0 – bcppipe7，保证在每一个进程中，可以向独立的管道串行的写入。
3. 为了不影响到原有的其它OPEN CLIENT应用，决定不升级当前的 OPEN CLIENT 11.1.1， 而是用OPEN CLIENT 12.0
4. 修改原调用BCP OUT进行批处理的 C 程序，使其使用OPEN CLIENT 12。BCP OUT的输出先写入管道，在由管道重定向至文件。

结果与结论：

1．经过修改的程序可以正确的BCP OUT出大于2G的文件。
2．由于采用管道方式做BCP OUT，性能略有下降。
3．需要注意，BCP IN 超过 2G 的大文件，也需要使用管道的方式。所以调用 BCP IN 的 C 程序也需要修改。

4.5 如何将 ASE SERVER 移植到同种平台（相同操作系统）的系统上

提示：

- N1 这里所列出的所有步骤，并不需要按顺序执行，只是表明了在实验中成功完成这项任务的顺序。
- N2 SYBASE 提示您保留直至今日的 ISQL 脚本，包括建立 login、create database、disk init 等等。使用这些脚本会使您完成这项任务更加容易。这些脚本也可以通过 master 中系统表的内容来重新生成，可能会繁琐一些。
- N3 成功的关键在于目标系统中系统表的字段与源系统中相应系统表的字段相同。特别是：syslogins 与 sysdatabases中的'suid 以及 'dbid';sysusages 中所有行的 segmap、lstart以及 size。

请在数据源系统上执行以下操作：

- 对所有的数据库进行数据一致性检查（DBCC），并后备所有的用户数据库。
- 保留 master 数据库中数据表的内容，使用 select * from table_name命令：

- **sysdevices,sysusages,sysdatabses**
- **syslogins,sysservers,syssvrroles,sysloginroles,sysremotelogins**

  对于 sysusages 表，请使用以下命令：

  select * from sysusages order by dbid,lstart

  对于 sysdatabases 表，请使用以下命令：

  select * from sysdatabases order by dbid

- **使用** bcp **命令拷贝（**2**）中所列出的系统表内容。**

  Unix：bcp master..table_name out file_name -Usa -Psa_password -c

  Vms：bcp master..table_name out file_name /username="sa"/sa_password/char

- **保留** sp_configure**命令的执行结果**

请在目标系统上执行以下操作：

- **安装并配置新的** ASE **和** Backup Server**。**

  确认所指定的 master、tempdb、sybsystemprocs **大小至少等于数据源系统上相应数据库的大小，同时确认与数据源系统相同的语言模块以及字符集。**

- **启动** ASE**，使之处于正常工作状态。参照数据源系统的配置（**4 **中保留的执行结果）修改目标系统配置与之相同，并确认**'device'**参数值至少等于源系统此参数值。**
- **在** model、sybsystemprocs**数据库中任意执行几个动作以判断数据库工作正常。请不要添加用户、角色、修改系统表。**
- **重启** ASE **以测试新配置有效。**
- **执行以下操作：**

```
1> use master
2> go
1> sp_configure "allow updates",1
2> go
```

  重启 ASE**。**

- 使用 bcp 命令拷贝（2）中所列出的系统表内容。

  Unix：bcp master..table_name in file_name -Usa -Psa_password -b 1 -c

  Vms：bcp master..table_name in file_name

  /user="sa"/sa_password/char/batch=1
- 建数据库设备，大小至少等于源系统中相应数据库的大小。
- 运行 create database 和 alter database 的脚本（或者使用命令行）。注意 create、alter顺序要与源系统 create、alter顺序一致，并使用与之相同的参数。完成后请对比源系统与目标系统中的 sysdatabases、sysusages，使之完全相同，否则要重新做 12 这步工作。（请参照 N3 提示）注意：
  - 在 10.0以及更高版本中 segmap 字段在做了数据库 load 之后会被修改。
  - 在确定需要相同的 dbid 时，则要采用与在源系统中 create、alter 相同的顺序，在目标系统中做 create、alter，并且使用相同的参数值。而这种需求仅仅是当数据库中某些 objects 要参考不同的数据库中的 objects 才会采用的。另外，这种需求只有在每一个数据库中都被采用，完成的结果才会使得 sysusages表中的 segment、lastart、size 字段，或者是 fragment 与源系统中数值相同。运行以下命令，与原系统的输出进行比较：

    select * from sysusages order by lstart
- load 用户数据库并执行 dbcc 检测。
- 执行以下操作：

  1> sp_configure "allow updates",0

  2> go

  重启 ASE。

后备 master库以及用户数据库。

4.6 如何扩展 master数据库空间

master数据库只能扩展在 master 设备上.那么当 master 设备已经没有足够的空间可使用时,请按以下步骤操作：

(此操作过程是以 UNIX 操作系统为例.SYBASE 安装路径为/sybase)

1.备份master 数据库

启动backup server，进入isql 环境执行：

```
1>dump database master to '/sybase/master.dump'
2>go
```

shutdown SQL/ASE Server

```
1>shutdown
2>go
```

2.创建新的足够大的master 设备

$buildmaster -d<master_device> -ssize(size 以2K 为单位)

例：$buildmaster —d/sybase/data/master.dat -s102400

3.修改RUN_servername 文件

编辑RUN_server_name 文件，-d 参数指向新建的设备名。

4.单用户模式重启server

$startserver -f RUN_servername -m

5.执行installmaster 脚本

6.由备份文件装载master 数据库

```
1>load database master from '/sybase/master.dump'
2>go
```

7.修改sysdevices 信息

```
sp_configure 'allow updates', 1
  go
  begin tran
  go
update sysdevices set high = 102399 , phyname =
'e:\sybase\data\master_test.dat' where name = 'master'
```

go

(102399=200*512-1 master 设备大小为 200M)

commit tran

go

8.扩展master 数据库

1>alter database master on master 设备名称=size(此值以 M 为单位)

2>go

例：alter database master on master=10

将 master 数据库在 master 设备上扩展 10M

4.7 Invalid tdslength value

```
Error such as:
00:00000:00000:2001/03/22 16:10:07.80 kernel  ksmask__rpacket: Invalid
tdslength value 21536, kpid: 1310740
00:00000:00000:2001/03/22 16:10:20.87 kernel  ksmask__rpacket: Invalid
tdslength value 21536, kpid: 1376277
00:00000:00000:2001/03/22 16:10:51.27 kernel  ksmask__rpacket: Invalid
tdslength value 21536, kpid: 1441814
00:00000:00000:2001/03/22 16:15:38.22 kernel  ksmask__rpacket: Invalid
tdslength value 21536, kpid: 1507351
```

It can be a heavy network traffic. Indicated that this message is an informational message that comes from the network. Possible causes may be: the size of the packet that the server has received is different to the size that was sent by the client, ksmask__rpacket: The SQL Server does not validate the size of incoming TDS network packets. Bad incoming data may cause waiting processes to hang as the server waits for a very large amount of data on a network socket. When a corrupt TDS packet has been received, due to tcpip sending begining of another TDS packet before finishes sending the current TDS packet, the server terminates the process and outputs the above message. Kernel error "ksmask__rpacket: Invalid tdslength" indicates that tcp/ip has sent a corrupt TDS packet to sqlsvr.

Troubleshooting the error:
-max network packet size
-additional network memory
-any changes made to network recently

Increasing the network memory could be a solution. I can think of a situation where there is no memory left so the server is unable to store the new packets. This could lead to invalid tdslength. There are some cases where increasing network packet size and the additional

network memory solved the problem.
In other cases, no further action was required as the message did not re-appear.
You can use **sp_configure** to increase the max network packet size and **additional network memory**, and also check if there's any issue with the network.
max network packet size:**取决于应用中需要发送的数据包的大小。**
additional network memory= max network packet size+ max network packet size*0.02
(**并且使此数值为** 2048 **的倍数。**)

4.8 如何将 master设备从 UNIX 的文件系统移到裸分区

1. Turn off asynch io

```
1> sp_configure "allow sql server async i/o", 0
2> go
You will need to reboot your server for this to take affect.
2. Mirror the master device to the new raw partition.
1> disk mirror name = "master",
2> mirror = "absolute path to new master raw partition"
3> go
2. Unmirror master (this will permanantly break the mirror to the master
device (If you want to expirement, you can set mode = retain and remirror
a few times. See Reference Manual for syntax).
1> disk unmirror name = "master",
2> side = "primary",
3> mode = remove
4> go
3. Shut down your SQL Server. In your run server file, there is a "-d"
flag that has the path to your old master device. Change this to the path
for the new master device.
4. Restart your server and turn on asynch io.
1> sp_configure "allow sql server async i/o", 1
2> go
5. Shutdown and restart your server one more time to enable the
asynch io. Verify that your server is using asynch io by looking at the
start up sequence in the errorlog.

NOTE: It is best to perform this task with the server in single user mode.
You can do this while you recycle your server to disable asynch io in step
1
above by adding a "-m" to your run server file. Be sure to remove the flag
when you are finished.
```

4.9 如何生成 bcp 命令文件(以 pubs2 为例)

本文适应于 isql 11.*，可通过 isql -v 得到版本

- **编辑一个文本文件** select.sql，**内容如下：**

```
set nocount on
use pubs2
go
select "bcp pubs2.." + name + " out " + name + ".bcp -Usa -P -c
"
from sysobjects where type="U"
go
```

- **如果是** unix, **执行：**

```
isql  -Usa   -P  -b -i select.sql  -o bcpout
chmod  +x   bcpout
```

- **如果是** Windows, **执行：**

```
isql  -Usa   -P  -b -i select.sql  -o bcpout.bat
```

将 select.sql **中的** out **换为** in, **重复以上步骤即可得到** bcp in **的命令文件**

4.10 如何动手修改 interfaces 文件

在一些 HP **和** SUN **的机器上，**interfaces **文件中关于** SERVER **的信息是以** 16**进制的形式存储的，必须要通过实用程序** dscp **才能进行修改。**

实际上，我们只要了解了这些 16**进制数据的格式，也可以直接通过** vi **来更改** interfaces **文件。**

下面以 e3000 **为例，介绍一下** interfaces**文件的结构和格式：**

用 vi **打开**/opt/sybase/interfaces，**可以看到这些信息：**

E3000

master tli tcp /dev/tcp \x00021a0a9e4d51f80000000000000000
query tli tcp /dev/tcp \x00021a0a9e4d51f80000000000000000

其中：

/x0002 : **保留字，不必修改**
1a01 : 16 **进制端口号，高位在左边，转换成** 10**进制为：**6666

9e4d51f8：16进制主机地址，

转换成 10进制为： 9e – 158
4d -- 77
51 -- 81
f8 – 248

即为： 158.77.81.248

可以修改的信息其实主要就是主机地址和端口号，所以，如果需要，只要按照以上格式修改其中信息即可。

4.11 关于 tempdb 的优化

缺省情况下，tempdb 数据库是放置在 master 设备上，容量为 2M，而临时数据库是活动最为平凡的数据库常常被用来排序、创建临时表、重格式化等操作，所以 tempdb 的优化应该受到特别的关注。

第一步：将临时数据库与高速缓冲进行绑定。

由于临时表的创建、使用，临时数据库会频繁地使用数据缓存，所以应为临时数据库创建高速缓存，从而可以使其常驻内存并有助于分散 I/O：

1、创建命名高速缓存

```
sp_cacheconfig “tempdb_cache”,”10m”,”mixed”
```

2、重新启动 server
3、捆绑临时数据库到 tempdb_cache 高速缓存

```
sp_bindcache “tempdb_cache”, tempdb
```

4、若有大的 I/O，配置内存池

第二步：优化临时表

大多数临时表的使用是简单的，很少需要优化。但需要对临时表进行复杂的访问则应通过使用多个过程或批处理来把表的创建和索引分开。以下两种技术可以改善临时表的优化

1、在临时表上创建索引

1） 临时表必须存在
2） 统计页必须存在（即不能在空表上创建索引）

2、把对临时表的复杂的使用分散到多个批处理或过程中，以便为优化器提供信息

下面的这个过程需要进行优化：

```
create proc base_proc
as
select * into #huge_result from auths
select * from article, #huge_result where article.author_code=
#huge_result.author_code and sex=”0”
```

使用两个过程可以得到更好的性能

1)

```
create proc base_proc
as
select *
into #huge_result
from auths
exec select_proc
```

2)

```
create proc select_proc
as
select * from article,#huge_result
where article.author_code=#huge_result.author_code and sex=”0”
```

说明：在同一个存储过程或批处理中，创建并使用一个表时，查询优化器无法决定这个表的大小。

4.12 ASE12.5.x 的一条有用的命令: disk resize

syntax:

```
disk resize
name='device_name',
size=additional_space
```

After using this command,you could alter database on the device that you just resize
to add dev additional_space .Then we need not use command of 'disk init'.

4.13 如何更改字符集为 cp936

(这里 SYBASE 的安装路径为 c:\sybase)

```
1.c:\>cd \sybase\charsets\cp936
2.c:\sybase\charsets\cp936> charset -Usa -Psa_pass -Sserver_name binary.srt
```

cp936
3.在SQL 环境中
```
1>select name,id from syscharsets
2>go
```
找到name 为 cp936 对应的 id(假设为 117)
```
4.1>sp_configure "default character set id",117
2>go
```
5.重启server 两次
（注:第一次启动后，server 会自动宕掉，需要第二次重启后才能使用）

5. 灾难恢复篇

5.1 数据库服务不能启动

5.1.1 如何查找数据库启动失败原因

在实际环境中，数据库 Server **无法启动的原因很多，本文仅列出了几种常见的情况，供您参考。**

首先，应检查 Server**的日志文件。不同版本缺省的日志文件如下（其中** <server_name>**为** Sybase Server **的名称：**
UNIX:

11.0*: $SYBASE/install/errorlog
11.5* 或 11.9*: $SYBASE/install/<server_name>.log
12.0*:$SYBASE/ $SYBASE_ASE/install/<server_name>.log

NT:

11.0* 11.5* 或 11.9*: %SYBASE%\install\errorlog
12.0*: %SYBASE%\ASE-12_0\install\errorlog

- **CASE 1:**
  basis_dlock: file '/sybase/master.dat' already in use by a SQL Server
  kernel kdconfig: unable to read primary master device
  kernel kiconfig: read of config block failed
  检查 server是否已经启动(showserver, ps -u sybase 或 NT service)
- **CASE 2:**
  dopen: open '/sybase/master.dat' failed, permission denied
  kernel kdconfig: unable to read primary master device
  kernel kiconfig: read of config block failed
  检查 master **设备文件的所有者及权限**

- CASE 3:
  ninit: All master network listeners have failed. Shutting down
  检查 network ip **及** port **配置** (netstat -a)

- CASE 4:
  kernel:kscsinit: connectivity library error. Operation: cs_ctx_alloc().
    - **检查操作系统参数是否已经修改并重新启动操作系统** (**参见安装手册**)

若操作系统异常宕机，ASE**未启动而** $SYBASE/<server_name>.krg **已经存在**, **删除该文件**

5.1.2 简单故障及排除方法

- **服务器启动失败，报告不能创建共享内存（UNIX 平台）：**
  解决方法：
  检查 Sybase **主目录下的文件，将所有文件的所有者改为** sybase **帐户。**
  编辑 sybase **主目录下的名为** *ServerName*.cfg**文件，找到** total memory**的一行，检查其配置值是否过大。**
  重新启动数据库服务器。
- **服务器启动失败，报告不能建立网络监听（UNIX 平台）：**

解决方法：

编辑 Sybase 主目录下的名为 interfaces 文件，检查相应服务器的网络端口是否和其它的服务器重复或使用的是系统已经使用的端口；

检查/etc/hosts 的主机名和 IP 地址，对照 interfaces 中的主机名，是否存在冲突。

重新启动数据库服务器。

- 服务器启动失败，报告不能打开 master 设备文件（UNIX 平台）：

  解决方法：

  检查$SYBASE/$SYBASE_ASE/RUN_*ServerName* 中的 master 设备文件的位置；

  检查 master 设备文件路径及访问权限；

  重新启动数据库服务器。

5.1.3 master数据库日志满了,server无法启动,怎么办 (error 1105)

```
[ERROR]
Can't allocate space for object 'syslogs' in database 'master' because the
'system' segment is full.
If you ran out of space in syslogs, dump the transaction log. Otherwise,
use ALTER DATABASE or sp_extendsegment
to increase the size of the segment.
[WORKAROUND]
1.在 RUN_servername 的文件中添加标识 : -T3607
(编辑 RUN_servername 文件,在文件末尾添加以上标识)
2.启动 ASE
(要直接运行文件 RUN_servername,且该运行窗口在执行以下操作时不能关闭)
3.截断日志
1>dump tran master with no_log
2>go
4.停止 ASE 服务
1>shutdown
2>go
5.删除 RUN_servername 文件末尾的标识 : -T3607
6.重新启动 ASE
```

5.1.4 SQL Server 10.x配置内存过大、Server不能启动时怎么办？

SQL Server 使用的内存与机器的内存总数有一定的比例关系。如果 Server 使用内存太小，影响到 SQL Server 的性能，但内存配置过大超过一定比例时，导致 SQL Server 不能启动。

在 ISQL 中，用 sp_configure “memory”可以看到你的 Server 现在使用的内存大小。这个数字单位为 Page，每一个 Page 为 2K。参数 memory 在启动 Server 时被读入内存。所以 memory 被修改之后，必须 shutdown Server，再重新启动 Server，新的参数才生效。

```
1>sp_configure "memory",number_of_memory
2>go
```

```
1>reconfigure with override
2>go
1>shutdown with nowait
2>go
```

如果新的内存配置太大，SQL Server 不能启动，那么必须修改 Master 设备上的内存配置参数。

在$SYBASE/install 目录下，使用 buildmaster 命令

```
$buildmaster -d/path/master.dat -y cmemsize = XXX
```

其中 XXX 为内存页数。如果新的内存大小合适，SQL Server 可以启动。

5.2 数据库被挂起

5.2.1 如何解决数据库被挂起的问题(error 926)

现象：Error 926
Severity Level 14
Error Message Text
Database 'xx' cannot be opened - it has been marked SUSPECT by recover Explanation

（1）当你使用 Transact_SQL 命令操作这个数据库的数据时，出现这个信息，这是一个严重的错误，如果你要使用这个数据库的数据，必须改正这个错误.

（2）启动 Backup Server，后备 master 数据库

```
1>dump database master to "/usr/sybase/master.dup"
2>go
```

（3）用 isql 登录到 ASE，须用 sa 帐号（本文以 pubs2 数据库为例）

```
1>sp_configure "allow updates", 1
2>go
1>begin tran
2>go
1>use master
2>go
1>update sysdatabases
2>set status = -32768
```

```
3>Where name="pubs2"
4>go
```

如果得到(1 row affected),则

```
1>commit
2>go
```

否则

```
1>rollback
2>go
```

(4)重新启动 ASE.

注：ASE 重新启动之后，当发现数据库本身存在不可恢复的问题时，如数据页损坏等，且没有完好的数据库备份，一定要用 bcp...out 备份用户数据库数据。此时，以下步骤省略，并按照"如何删除坏的用户数据库"文章删除此数据库。之后重建此数据库，恢复备份。

否则，按以下步骤继续操作：

用 sa 帐号注册到 ASE.

```
1>begin tran
2>go
1>use master
2>go
1>update sysdatabases
2>set status=0
3>Where name="pubs2"
4>go
```

如果得到(1 row affected),则

```
1>commit
2>go
```

否则

```
1>rollback
2>go
```

```
1>sp_configure "allow updates" ,0
2>go
```

(5)重新启动 ASE.

(6) 如果你的数据库原来有 dboption(例如"select into","trunc log on chkpt"等), 你需要重新设置这些 option.

(7) 当数据库已经恢复可使用状态后，运行 dbcc 命令检查数据库的一致性(参照“如何检查数据库中数据一致性”文章)

(8) 备份用户数据库

例如：

```
1>dump database pubs2 to "/usr/sybase/pubs2.dup"
2>go
```

5.2.2 如何做 rebuild log

注意：这个过程可能会引起数据的不一致性。

(1)赋予 sa 用户 sybase_ts_role 的角色

```
isql -Usa -P
1>sp_role "grant","sybase_ts_role",sa
2>go
1>quit
```

(2)将数据库置为"bypass recovery"状态

```
isql -Usa -P
1>sp_configure "allow updates",1
2>go
1>use master
2>go
1>update sysdatabases set status=-32768
2>where name="database_name"
3>go
```

```
1>shutdown with nowait
2>go
```

(3)rebuild 数据库日志

```
isql -Usa -P
1>use master
2>go
1>dbcc rebuild_log(database_name,1,1)
2>go
1>shutdown with nowait
2>go
```

(4)重启 ASE

```
1>use master
2>go
1>update sysdatabases set status=0 where
name="database_name"
2>go
1>sp_configure "allow updates",0
2>go
1>shutdown with nowait
2>go
```

(5)**在重启** ASE **之后**,**如果数据库恢复正常，**rebuild log **工作将会成功完成，否则要恢复数据库备份，使用** dump database **或** bcp **命令。**

5.3 数据库恢复

5.3.1 如何删除坏的用户数据库？(以pubs2为例)

当使用 drop database 无法删除数据库时，使用本文所示方法可以删除。

(1)使用 isql 以 sa 注册 ASE

(2)设置允许修改系统表

```
1>sp_configure "allow updates",1
2>go
```

(3)把 要删除的用户数据库置为"suspect"状态

```
1>use master
2>go
1>begin tran
2>go
1>update sysdatabases set status=256
2>where name="pubs2"
3>go
```

如果得到(1 row affected),则

```
1>commit
2>go
```

否则

```
1>rollback
2>go
```

(4)重启 server,并用 isql 以 sa 注册。

(5)删除数据库

```
1>dbcc dbrepair(pubs2,dropdb)
2>go
```

(6)恢复允许修改系统表

```
1>sp_configure "allow updates",0
2>go
```

(7)结束

5.3.2 如何做Rebuild Master

提示 1：
如果有可能，在执行这个任务之前，请先做操作系统级 SYBASE DEVICES **的后备。** UNIX **操作系统可使用命令**"dd"**。因为如果** disk reinit **使用了错误的信息，那么，在执行了** disk refit **之后就会产生无法弥补的错误。倘若，存在一个** SYBASE DEVICES **的后备文件，将会给我们一个弥补的机会。例如：当** disk reinit **使用了过小的** size **值，我们还可以重新恢复** SYBASE DEVICES **文件，重新做** disk reint、 disk refit。

提示 2：
在使用 disk reinit **命令时，将覆盖** SYBASE DEVICE**（请参照以下语法），安全的做法是** size **值使用裸分区或系统文件的大小的最大值。如果使用的是** UNIX **裸分区，即使你不能确认** SYBASE DEVICE **最初大小是不是最大值，都要使用裸分区大小的最大值。**
步骤：

- **获得将要被恢复的** SYBASE DEVICE **的信息。**
  这些信息被用来重建 sysdevices,sysusages **以及** sysdatabases。
  - **从** error log **的** server **启动信息中获得** SYBASE DEVICE **的设备名、指定路径。**
  - **使用裸分区或系统文件的大小的最大值作为** SYBASE DEVICE **的大小。**
  - **以上信息也可以通过最近的** sysdevices **系统表的内容来获得。如果对此信息怀有疑问，还是使用以上的方法比较稳妥。**
- **做操作系统级的** SYBASE DEVICE **后备。**UNIX **操作系统，使用**"dd"**命令实现。**
- **配置一个新的** ASE Server。**在以后的步骤中会用到这个新的** master。
- **如果需要，请再配置一个** Backup Server。

- 用单用户模式启动 Server。
- 运行 disk reinit，用来重建 sysdevices 系统表，而没有重新初始化 SYBASE DEVICE。

  语法如下：

  disk reinit
  name="device_name",
  physname="physical_name",
  vdevno=virtual_device_number,
  size=number_of_blocks

  完成后，请查看 error log。
- 确认重建的 sysdevices 系统表中信息正确：

  ——select * from sysdevices

  比较表中的信息是否与 error log 或者保留的 sysdevices 中的信息相同。
- 运行 disk refit,用来重建 sysdevices 以及 sysdatabases。

  用法如下：

  use master
  go
  disk refit
  go

  查看 error log 中是否有错误提示。
- 当 disk refit 完成后，会自动 shut down ASE Server。
- 确认重建的系统表的信息是否正确：

  ——单用户模式启动 ASE Server

  ——select * from sysusages、select * from sysdatabases

  ——sysusages 系统表看起来是否正确？可以和以前保留的的信息进行比较。如果没有这种可能，那么应该保证不出现显而易见的错误。例如：是否缺少 dbid；是否缺行；是否对于一个数据库来说只有 segmap=4(表示为日志行)的行等等。

  ——sysdatabases 系统表看起来是否正确？是不是没有显而易见的错误？

  11、 启动 ASE Server，查看是否所有的数据库都已经正常恢复。
- 对所有的库做 dbcc 检查。
- 对所有库做后备。

请参考《Trouble Shooting Guide》

5.3.3 如何恢复 master数据库

ASE can't setup and has no valid dump of master

1、编辑 RUN_servername

在命令行最后加入：-T3607

2、单用户模式启动 ASE

```
$cd install
$startserver -f RUN_servername -m
```

3、bcp out 系统表

```
$bcp master..sysdevices out /directory.spec/devs -Usa -P -c
$bcp master..sysdatabases out /directory.spec/dbs -Usa -P -c
$bcp master..sysusages out /directory.spec/usages -Usa -P -c
$bcp master..syslogins out /directory.spec/logins -Usa -P -c
$bcp master..sysconfigures out /directory.spec/configures -Usa -P -c
$bcp master..syscharsets out /directory.spec/charsets -Usa -P -c
```

4、shutdownASE

5、创建新 master 设备

```
$buildmaster -d<path_to_new_master_device> -s<new_master_device_size>
```

(new_master_device_size 以 2K 为单位)

6、编辑 RUN_servername

将指定 master 设备指定为新创建的 master 设备，并删除在第 1 步中增加的参数。

7、删除/directory.spec/dbs、/directory.spec/usages 文件中有关 master、tempdb、model 的内容。

8、单用户模式启动 ASE

```
$cd install
$startserver -f RUN_servername -m
```

9、 bcp in 系统表

```
$ bcp master..sysdevices in /directory.spec/devs -Usa -P -b 1 -c
$bcp master..sysdatabases in /directory.spec/dbs -Usa -P -b 1 -c
$bcp master..sysusages in /directory.spec/usages -Usa -P -b 1 -c
$bcp master..syslogins in /directory.spec/logins -Usa -P -b 1 -c
$bcp master..sysconfigures in /directory.spec/configures -Usa -P -b 1
-c
$bcp master..syscharsets in /directory.spec/charsets -Usa -P -b 1 -c
```

10、shudown ASE

11、执行 installmaster 脚本

```
$isql -Usa -P < $SYBASE/scripts/installmaster
```

12、启动 ASE

5.3.4 如何移植master设备

以下步骤说明了如何将 master 设备移植到不同的磁盘上。在执行此项任务的同时，请参看 SYBASE 的相关文档（Technical Documents#1324 entitled "Segment Remapping with Load database When Moving a Database"）。请注意，在执行 buildmaster 之前，要停止 ASE 的应用。

成功的关键在于，新建的 sysusages 系统表中每一行内容与旧的 sysusages 系统表内容相符。

- 对 master 库做 dbcc 检查，并后备 master 库。
- 执行 select * from table_name命令，并保留其输出内容。其中，table_name包括：sysdevices、sysusages、sysdatabases。同样，可以使用 bcp 命令来实现。
- 执行 sp_configure 命令，并保留其输出内容。
- 拷贝$SYBASE/server_name.cfg 文件，以做保留。
- Shut down SQL/ASE Server。

- **执行以下命令，创建一个新的** master **设备：**

  UNIX：buildmaster -d<master_device> -ssize

  VMS：buildmaster /disk=<master_device> /size=size

  (size **以页为单位，**1 **页**=2K)

- **编辑** RUN_server_name **文件**,-d(UNIX)**或**/device(VMS)**参数指向新建的设备名。**

- **单用户模式启动** SQL/ASE Server**：**

  UNIX：startserver -f RUN_sever_name -m

  VMS：startserver /server=server_name /masterrecover

- **执行** select * from sysdevices **命令，并保留其输出内容。**

- **确认新建的** sysusages **系统表中每一行内容与旧的** sysusages **系统表内容相符，而且在配置好** Server **之后，没有做过** alter database**，那么系统表** sysusages **的内容是正确的。如果做过** alter database,**则要按原来的顺序执行这些脚本，如果没有脚本，就要到保存的** sysusages **系统表的信息中找到** alter database **的参数。**

- Shutdown SQL/ASE Server**，并用单用户模式启动** Server**，查看** sysusages **系统表内容是否正确。**

- **如果所配置的** Backup Server **名称不是** SYB_BACKUP,**则要执行：**

  1> sp_configure "allow updates",1

  2> go

  1> update sysservers set srvnetname="name in interfaces file"

  where srvname="SYB_BACKUP"

  2> go

- **装载** master **库。如果新** master **设备的大小与旧设备大小不同，则** Server **会** Shut down**。请注意新的系统表将会被重写，而且你需要调整** Sysdevices **系统表中** master **设备的大小。请执行以下两步：**

- **用单用户模式启动** Server.

- **如果新设备大小与旧设备大小不同，请执行：**

  1> sp_configure "allow updates",1

  2> go

  1> update sysdevices set high=nnnn where name="master"

  2> go

  其中 nnnn **是以页（**2K**）为单位的** master **设备的大小，此值可以从保留的** sysdevices **系统表的信息中找到。如果所建的设备比旧设备大，请执行：**1**）创**

建一个与 master 设备差不多大的数据库，这样做的目的是重新初始化分配页，使得整个 master 设备可用。2）删除这个数据库。

- 重启 Server。
- 后备 master 数据库。

注意：

- 在装载 master 库之前要确定 ASE 此时的字符集和语言模块与后备 master 库时的字符集和语言模块相同。
- 在执行此任务之前和完成以后使用 sp_helpsort 查看字符集与语言模块是否相同。

5.3.5 如何重建sybsystemprocs系统数据库

依照以下步骤可以实现移动 sybsystemprocs 系统数据库以及设备的任务。同时这个过程也可以用来扩建 sybsystemprocs 系统数据库。

以下过程中所提到的语法结构，可以参看 SYBASE 相关资料。

SYBASE 提醒您，在修改系统表时，ASE 要以单用户模式运行，同时要以"sa"用户登录。

- 保留驻留在 sybsystemprocs 系统数据库中自定义的存储过程脚本。
- 单用户模式启动 ASE，执行：

```
1> sp_configure "allow updates",1
2> go
1> reconfigure with override(10.0版本以上，省略此步)
2> go
```

- 删除 sybsystemprocs 系统数据库:

```
1> use master
2> go
1> drop database sybsystemprocs
2> go
```

在重建 sybsystemprocs 系统数据库之前，不要创建任何其他数据库。

- 删除 sysdevices 系统表中有关 sybsystemprocs 系统数据库的信息：

```
1> begin tran
2> delete from sysdevices where name="sysprocsdev"
```

我们假定 sysprocsdev 是默认的 sybsystemprocs 系统数据库设备名。

3> select * from sysdevices

确定删除是否正确有效，如果正确，执行：

4> commit tran

否则，执行：

4>rollback

- **重启** ASE.
- **创建** sybsystemprocs **系统数据库设备：**

  1> disk init name="sysprocsdev",physname="physical_path",vdevno=4,size=25600

  2> go

  其中 size **是以页为单位（**512 **页**=1M**）。**

  sybsystemprocs **系统数据库设备的** vdevno **应该是** 4**，**ASE **在单用户模式下** 4 **不能被重新利用，所以如果以上语句执行时出现问题，请重启** ASE**。**

- sybsystemprocs **系统数据库：**

  1> create database sybsystemprocs on sysorocsdev=50

  2> go

  在 sysdatabases **系统表中，**sybsystemprocs **系统数据库的** dbid=4**，如果在重建** sybsystemprocs **系统数据库之前，没有创建任何其他数据库。**

- **以多用户模式启动** ASE, **并执行：**

  1> sp_configure "allow updates",0

  2> go

  3> reconfigure with override(10.0**版本以上，省略此步**)

  4> go

- **运行** installmaster **脚本：**

  %isql -Usa -Psa_password -Sserver_name -n -iinstallmaster -o< output_file

  installmaster **脚本在**$SYBASE/scripts **路径下。**

- **重建** sybsystemprocs **系统数据库中的用户自定义存储过程。**

5.3.6 如何挽救 corrupt table 中的数据

[**此文仅供参考**]

***/设置数据库隔离级别：**
sp_setsuspect_granularity [dbname[,{"database"|"page"}[,"read_only"]]]
强制脱机数据库/数据页联机：
sp_forceonline_db dbname,{"sa_on"|"sa_off"|"all_users"}
sp_forceonline_page dbname,pgid,{"sa_on"|"sa_off"|"all_users"}
获取有关脱机数据库/页的信息：

```
sp_listsuspect_db
sp_listsuspect_page [dbname]
```

How to patch a corrupt table and save its data

platform: general product: ASE
written by: Hunter Liu
last update time : Dec.17,1998

(1) isql -Usa -P to log into sql server and enter following commands

```
1> sp_role "grant","sybase_ts_role",sa
2> go
1>quit
2>go
```

(2) relogin into sql server with sa

(3) enter these commands in isql:

```
1> use your_database
2 > go
1> select first ,root ,indid from sysindexes
2> where id=object_id("table_name")
3>go
```

choose a line from the output of which the indid is neither 0 nor 1,
and pick up the value of its root. this is the root page number
of this index btree,next we will use root_page_number
to represent it and indid for its index id.

(4) find a page number which is on the data pages chain

```
1>dbcc prtipage(database_name,table_name, indid,root_page_number)
2>go
```

(4.1) read the last line of this dbcc output ,it is like:
index row at offset XX points to page XXXX.
take the page number xxxx and issue above command again :

```
1>dbcc prtipage(database_name,table_name,indid,XXXX)
2>go
```

repeat this step until the dbcc output looks like this:
leaf row at offset xxxxx points to data page nnnnn,row number xxxx.
this meens that we reach the leaf-level pages of this index btree,
so you choose a line from its output and read out the
page number, which is a data page of this table.

(5) from the data page (nnnn), you walk along the data page chain of this table
until we get to the begin, the first page of its database chain.

```
1>dbcc pglinkage(supportdb,nnnn,0,2,0,0)
2>go
```

the last number of this page number list is the first
page of this table. make sure the object id displayed is correct.

(6) update the first column of this table in sysindexes

```
1>update sysindexes set first=new_first_page where
id=object_id('table_name')
2> and indid=0
3>go
```

(7) use select count(*) from table_name to test our work.

(8) select * into new_table from table_name.

(9) drop the old corrupt table.

5.3.7 Recovering the master Database or Master Device under ASE 12.5

This TechNote describes the new procedure you need to use to load the master database, or recover from master database or device corruption in Adaptive Server Enterprise 12.5. The disaster recovery procedures differ from pre-12.5 servers.

Contents

Section 1. Background

Starting with Adaptive Server version 12.5, there is no **buildmaster** program to build the master device. Rather, this functionality has been incorporated into the **dataserver** (unix) and **sqlsrvr** (Windows) programs. The server now allows you to create master devices and databases with 2K, 4K, 8K or 16K logical page sizes. Due to these and other changes, you cannot use the instructions provided in the *ASE Troubleshooting and Error Messages Guide* (EMTSG) for disaster recovery tasks like recovering the *master* database or device.

This TechNote explains how to perform disaster recovery in the 12.5.x server. It is applicable to ASE 12.5.0.1 IR and higher versions.

Note:

The EMTSG instructions still apply to the pre-12.5 servers.

Section 2. Before You Begin

This document describes three key maintenance and disaster recovery tasks:

- Loading an older copy of your *master* database. This assumes that both the master device and *master* database are intact and free from corruption.

- Recovering from a corrupted *master* database. This assumes that the master device is intact.

- Recreating a master device and all its databases.

Some points to note before using this information:

- This material applies only to ASE 12.5.x and higher. It has been verified with ASE 12.5.0.1 Interim Release (IR), and it is recommended that you use the procedures with this release (or later). For 12.0 and prior versions, use the instructions provided in the *ASE Troubleshooting and Error Messages Guide*, "System Database Recovery."

- This material assumes that your Adaptive Server was installed with your platform's default sort order. If you have installed a non-default sort order, you must ensure that your (restored) server uses the correct sort order and character set to reflect that in the dump. Refer to the *ASE Troubleshooting and Error Messages Guide*, "System Database Recovery," section titled "Valid Dump with Non-Default Sort Order" for details; and note that in Step 3 of that section, the **sybinit** utility has been replaced by the **dsedit/dscp** utilities.

- All SQL command examples in this document use Transact-SQL syntax. All command-line examples are unix commands; Windows users can find the equivalent NT syntax in *ASE Utility Programs for Windows and Windows NT*.

- Starting with 12.5, the **dataserver** command allows a space between option and parameter.

- The examples in this document assume that

  - the **dataserver** binary is located in *$SYBASE/bin/dataserver*
  - the master device is *$SYBASE/d_master.dat*

  Replace this location and device name with those appropriate for your site.

- The server must be in single-user mode (that is, started with the **-m** flag) to load the *master* database. In this mode only the *master* database can be loaded.

- You can only load a dump of *master* that matches your server level. Loading an older version dump to a newer server is not permitted.

- You can use these procedures regardless of which version of the server you were using when you created your old master device. The server will find and correct any placement differences between the old and recreated databases.

- After the load completes but before shutting down, the server does some post-processing to reconcile the newly loaded *sysdatabases* and *sysusages* tables against the information in the master device. At this time the server may print a variety of error messages regarding failures to use or find the *master* database, and/or attempts to insert duplicate keys or duplicate rows to *sysusages* . Ignore these messages; they occur only during the reconciliation phase, and will not affect the server's operation after it shuts down and is restarted.

Section 3. Loading an older copy of *master* database

Use the following steps if your *master* database and the master device are intact, and you simply wish to load an older dump of your *master* database.
Note:
Be sure to read Section 2, Before You Begin.
Step 1: Put the Server in Single-user Mode
Shut down and restart the server with the **-m** flag, which places the server in single-user mode and sets up to load the *master* database:
% $SYBASE/bin/dataserver -d $SYBASE/d_master.dat -m
Step 2: Establish the Backup Server Name
This step is necessary to ensure that Adaptive Server has access to its backup server for dumps and loads. Follow the instructions detailed in Section 6, Manually Setting the Backup Server Name.
Step 3. Load the master Database
Issue the following **isql** command:

```
1> load database master from "master_db_dump"
2> go
```

Adaptive Server shuts itself down after the load is complete.

Section 4. Recreating the master Database

Use this procedure when the current master device is usable, but you are unable to use the server because of *master* database corruption. These steps enable you to create a new *master* database and reload it from backup.
Step 1. Create a New master Database
The approach to creating the new *master* database depends on the extent and nature of the corruption. Three different scenarios are possible:

- Basic recreation, which is sufficient if only the data in *master* was affected. The server reads the master device to determine page and device sizes.
- Recreating when the device's configuration area is corrupted. You will need to provide page and device sizing information.
- Recreating when the master database allocation pages are also corrupted. All corrupt or unallocated extents on the device are allocated to *master*.

Basic Recreation of master Database

This command instructs the server to read the device's configuration area to obtain page size and device size and determine where to place the *master* database:

% $SYBASE/bin/dataserver -d $SYBASE/d_master.dat -w master

The server creates a *master* of the same size, and in the same locations on disk, as the database it is replacing. It will NOT have the old database's data! Instead, it contains a default set of data that you will replace later via **load database**. The default data includes information about any databases existing on the master device (but no other devices). It also has minimal system information, including a login for *sa* with a *null* password.

This process produces a large number of "upgrade" messages tracking the progress of database creation which are helpful in troubleshooting any problems. They are "upgrade" messages because the server creates a new *master* database by "upgrading" the device.

Note:

If the configuration area is corrupt or unavailable, this command returns the message: "The configuration area in device 'xxx' appears to be corrupt. The server needs this data to boot, and so cannot continue." If this occurs, continue with the instructions below.

Recreation with a corrupt configuration area

The "Basic Recreation" process above may fail if the device's configuration area has become corrupt. If so, you must supply sizing information. You will need two parameters: the page size (you need to know what this was), and the device size, which you can determine directly from the device:

% ls -l $SYBASE/d_master.dat

Divide the size shown by the page size (2048, say) to obtain the number of server pages, by 1024 to obtain KB, or by 1048576 to obtain MB.

Provide this information on the command line as follows:

% $SYBASE/bin/dataserver -d $SYBASE/d_master.dat -w master
-z page_size -b device_size

For example, if your page size is 2K and the device size is 51204 server pages (100 MB, plus 8K space for the configuration area), the command looks like this:

% $SYBASE/bin/dataserver -d $SYBASE/d_master.dat -w master -z 2k -b 51204

You may also specify the device size as Kb, Mb, or Gb; for example, "-b 100M".

Recreation when master database allocation pages are corrupted

If the above procedures for recreating the *master* database fail, the database's allocation pages are corrupt. (This may happen, for instance, if the database device was inadvertently written over by a completely different file.)

In this case, you can force the server to allocate all corrupted or unallocated extents to the *master* database:

% $SYBASE/bin/dataserver -d $SYBASE/d_master.dat -w master -f

This allocates ALL corrupted or otherwise unrecognizable extents to the *master* database. Depending on the extent of your master device corruption, and how much free space it originally had, this will probably leave *master* much larger than it needs to be, causing it to occupy space that used to belong to other databases like *model, tempdb,* and *sybsystemdb*. We will consider recovering from that situation later.

Note:

You may combine the **-f, -b,** and **-z** options as necessary.

Step 2. Restart the Server in Single-user Mode

The server shuts down after recreating the *master* database. Restart it with the **-m** flag, which places the server in single-user mode and sets up to load the *master* database:

% $SYBASE/bin/dataserver -d $SYBASE/d_master.dat -m

Step 3: Account for Missing Databases (if you used the **-f** option)

Note:

You only need this step if you used the **-f** option in Step 1 to recreate the *master* database due to allocation page corruption. If you did not use **-f**, proceed to Step 4. Recall that the **-f** command line option could make the new *master* larger than needed at the expense of other required databases on the master device. You will need to check for these databases before proceeding. This step has many possible permutations, so you must know what databases *should* be on the master device to perform this step. For example, if you had moved *tempdb* to a different device, you will not need *tempdb* on the master device. If upgrading, you may well have created *sybsystemdb* on a device other than *master*; if so, you will not need to account for *sybsystemdb*.

Log in as *sa*, and check the databases currently on the master device:

1> select name from sysdatabases
2> go

Do you see all the databases that should be on the master device? If so, skip the rest of this step. Otherwise, you will need to determine which databases are missing *and* how big they should be, then obtain the free space needed to recreate these databases.

The following **isql** script obtains the required space by removing it from the end of the *master* database. In order, it

- establishes how many logical pages the missing databases need
- subtracts that number from the pages that *master* occupies
- removes disk usage entries for parts of *master* above that limit
- restricts the highest logical chunk of *master* such that its total size leaves the required number of pages free.

You will need to provide the required space value, denoted as "*@needed_mb*".

Note:

This sample script is provided to assist you with the disaster recovery task. It is not officially supported by Sybase.

```
1> declare @needed_mb int, @needed_pages int, @master_end int,
2>      @pgspermb int
3> select @pgspermb = (1048576 / @@maxpagesize)
4> select @needed_mb = 12 -- replace '12' with required space value
5> select @needed_pages = @needed_mb * @pgspermb
6> select @master_end = sum(size) - @needed_pages
7> from master.dbo.sysusages
8> where dbid = 1
9> if (@master_end > (6 * @pgspermb))
10> begin
11>     delete master.dbo.sysusages
12>     where lstart > @master_end
13>     update master.dbo.sysusages
14>     set size = @master_end - lstart
15>     where dbid = 1
16>     and lstart = (select max(lstart) from master.dbo.sysusages
```

```
17>              where dbid = 1)
18> end
19> else
20> begin
21>     print "Can't take enough space from the master database!"
22>     print "Need to find %1! pages", @needed_pages
23>     print "That would leave master with %1! pages", @master_end
24>     print "Cannot continue."
25> end
26> go
```

Note:

If the procedure fails, your master device is not big enough to hold all the databases you are trying to create. Check the required MBs of space that you specified. If it is correct, it may be necessary to create a new master device using the instructions in Section 5, Recreating the Master Device.

You now have enough space to recreate your required databases. Create them one at a time. For example:

1> create database model on default = 3
2> go

Repeat for each database. Then shut down the server, and restart it in single-user mode using step 2 above.

Step 4. Establish the Backup Server Name

This step is necessary to ensure that Adaptive Server has access to its backup server for dumps and loads. Follow the instructions detailed in Section 6, Manually Setting the Backup Server Name.

Step 5. Load the master Database

Issue the following **isql** command:

1> load database master from "master_db_dump"
2> go

Unlike during a normal database load, the server may need to perform a lot of extra work, because *master* contains information about the identity and location of your other databases; that information may have changed for this master device, and the server must check and update it as necessary.

At the end of this step, the server contains correct disk usage information about the master device. This may differ from the information in *sysusages* in the dump, so the server will find and correct both the size and location information for the databases. Any entries for parts of databases that don't actually exist on master will be removed.

During post-processing from the load, you may see one or more errors from the server. Read Section 1, Before You Begin, for more information about these errors.

After checks and validations are complete, the server will shut down. You may now restart it normally.

Step 6. Did you recreate any databases in step 3 above?

If you recreated databases using the procedure in step 3 above, load those databases. You must restart the server *without* the **-m** flag in order to accomplish this.

Section 5. Recreating the Master Device

Use these steps when the disk that used to contain your master device is not accessible, and you need to start over with a new device.

This situation is somewhat similar to the scenario in Section 4 above in which the master device becomes so corrupt that you need to use the **-f** option, because you will need to know what databases used to be on your master device and how big they were so you can validate and recreate them as necessary.

Step 1. Create your new master device

When creating the new master device, make sure you use the same page size as your old master device and make the new device at least as large as the old one. The following example creates a device with a 2048-byte logical page size, and total size 100 Megabytes + 8 Kilobytes (the 8 KB is extra space for the configuration area.)

% $SYBASE/bin/dataserver -d $SYBASE/d_master.dat -z 2k -b 51204

Use the **-s** option with this command to specify the server name. You can also specify the "-b" size in Kb, Mb, or Gb. In the above example you would use "-b 100.00782M". Without one of the K, M, or G modifiers, the default device size is expressed in server virtual pages, 2048 bytes each.

At device creation, the server issues large numbers of "upgrade" messages tracking its progress; these messages help troubleshoot any problems. They are upgrade messages because the server creates a new installation by doing an "upgrade" of a device that it has just created.

When finished, the server shuts down. You now have a *master* database containing minimal system information, including an *sa* login whose password is *null*, and minimally sized *master, model, tempdb*, and *sybsystemdb* databases.

Step 2. Put the Server in Single-user Mode

Shut down and restart the server with the **-m** flag, which places the server in single-user mode and sets up to load the *master* database:

% $SYBASE/bin/dataserver -d $SYBASE/d_master.dat -m

Step 3. Establish the Backup Server Name

This step is necessary to ensure that Adaptive Server has access to its backup server for dumps and loads. The new master database contains a default Backup Server entry of *SYB_BACKUP* for *srvnetname*, which is probably wrong. Since the **sp_addserver** procedure is not available at this time, log in to the server as *sa* and update *sysservers* directly:

1> update master.dbo.sysservers
2> set srvnetname = "backup_server_name"
3> where srvname = "SYB_BACKUP"
4> go
(1 row affected)

Step 4. Load the *master* Database

Issue the following **isql** command:

1> load database master from "master_db_dump"
2> go

The server inspects the master device and makes any corrections needed in the newly loaded *sysdatabases* and *sysusages*. These corrections affect only the master device, since that is the only device that changed – the server assumes that all your other devices are undamaged and need not be inspected.

After this step, it is possible that your new master device contains database entries for databases that also exist on other devices in your system. This may happen if you moved *tempdb* to a different device, or created *sybsystemdb* on a different device. The server recognizes and handles this situation: if it finds pre-existing

entries for those databases on other devices, it presumes that the existing entries are correct and does not change them.
During post-processing from the load, you may see one or more errors from the server. Please read Section 2, Before You Begin, for more information about these errors.
After the checks and validations are complete the server shuts down. You may now restart it normally.
Step 5. Check that the Databases on Master Device Are Correct
When you created a new master device in step 1 above, the server created only its default set of databases, with minimal data. You will almost certainly need to load dumps of the databases (notably *model*) that used to be there.
Are the databases on your new master device large enough to hold the dumps you will be loading into them? Are all the necessary databases present? Is there any obsolete data that you need to clean up?
Log in as **sa** and inspect the databases on your system:
```
1> declare @pgspermb int
2> select @pgspermb = 1048576 / @@maxpagesize
3> select "db name"=db_name(dbid), dbid, "size"=sum(size) / @pgspermb
4> from master.dbo.sysusages
5> group by dbid
6> go
```
This command shows you all the databases present on your system, and their total size. Note that the *size* column in the output is expressed in Megabytes.
Does this list contain any entries where database name is *null*? These *sysusages* entries don't have any matching entries in *sysdatabases* ; they are unnecessary and should be deleted. (You may be specially susceptible to this if you upgraded from pre-12.0 versions, and created *sybsystemdb* on the older version; *sybsystemdb* will have a different dbid than the default *dbid*.) To remove these entries, use a script like the following:
```
1> exec sp_configure "allow updates", 1
2> go
1> delete sysusages
2> where db_name(dbid) is null
3> go
1> exec sp_configure "allow updates", 0
2> go
```
Are any databases missing? Create those databases. Are the databases large enough? If not, alter them to be at least large enough to hold the dumps. (It is okay if they are too large; the server simply clears the excess space.)

Section 6. Manually Setting the Backup ServerName
This procedure updates the *sysservers* table and is needed to ensure that Adaptive Server can access the correct backup server to carry out dumps and loads. Use it with the instructions for Sections 3 and 4.
Execute the following **isql** commands in Adaptive Server:
```
1> use master
2> go
1> select srvname, srvnetname from sysservers
```

2> where srvname = "SYB_BACKUP"
3> go

There are three possible outcomes to this query. Take the appropriate action below depending on the outcome:

Outcome	Action
ASE returns a single row and *srvnetname* contains the correct Backup Server name	No action is needed.
ASE returns a single row but the *srvnetname* is not the correct Backup Server name	Issue the following **isql** command: **1> update sysservers** **2> set srvnetname = "backup_server_name"** **3> where srvname = "SYB_BACKUP"** **4> go** **where *backup_server_name* is the name of the Backup Server as it appears in the *interfaces* file.**
ASE returns 0 rows	Issue the following **isql** command: **1> sp_addserver SYB_BACKUP, null,** **2> backup_server_name** **3> go**

5.3.8 如何只用数据库设备文件生成新的数据库服务器

本例的环境：

OS: Win2000 professional (中文)
ASE: Adaptive Server Enterprise 12.0 for NT(中文)
原有数据库服务器： PDS_CJ
数据设备文件：c:\sybase\data\master_RDS_CJ.dat 30M
c:\sybase\data\sybprocs_RDS_CJ.dat 80M

目的：

将通过现有的 2 个数据库设备在相同的平台下的其他的机器上创建数据库服务器，而不用 syconfig.exe(服务器配置)程序来创建服务器。

新建服务器名称：RDS_CJ

步骤：

一、配置新的数据库服务器：

1。拷贝原有数据库设备文件到新的机器上：
copy master_PDS_CJ.dat 为 master_RDS_CJ.dat
copy master_PDS_CJ.dat 为 master_RDS_CJ.dat

2。拷贝或新建 RUN 文件：
copy RUN_PDS_CJ.bat > RUN_RDS_CJ.bat
编辑 RUN_RDS_CJ.bat 将所有 PDS_CJ 替换为 RDS_CJ（指定 MASTER 设备和 SERVER 名）

RUN 文件样本：
```
rem
rem Adaptive Server Information:
rem name: PDS_CJ
rem master device: c:\sybase\data\master_PDS_CJ.dat
rem master device size: 15360
rem errorlog: c:\sybase\ASE-12_0\install\errorlog_PDS_CJ
rem interfaces: c:\sybase\ini
rem
c:\sybase\ASE-12_0\bin\sqlsrvr.exe
-dc:\sybase\data\master_PDS_CJ.dat
-sPDS_CJ
-ec:\sybase\ASE-12_0\install\errorlog_PDS_CJ
-ic:\sybase\ini -Mc:\sybase\ASE-12_0
```

3。添加 ASE 的服务器明：
用 dsedit 工具添加接口（interfaces）文件的条目

4。启动新配的服务器：（此时系统存储过程库仍无法使用）
通过执行 RUN 文件的方式启动服务器。(-T3604)

5。修改数据库中的信息：
```
1> select * from sysdevices where name="sysprocsdev"
2> go
low high status cntrltype name phyname mirrorname
---------------------------------------------------------------------------------------
16777216 16818175 16386 0 sysprocsdev c:\sybase\data\sybprocs_PDS_CJ.dat
NULL

1> sp_configure "allow update",1
2> go
1> update sysdevices set phyname="c:\sybase\data\sybprocs_RDS_CJ.dat" where
```

```
name="sysprocsdev"
2> go
(1 row affected)
1> update sysservers set srvname="RDS_CJ" where srvname="PDS_CJ"
2> go
1> shutdown
2> go
```

6. 启动服务器
   通过执行 RUN 文件的方式启动服务器(不加 -T3604 了)，新服务器配置完成。

二、将数据库服务器注册到注册表中：

1。编辑 Server(PDS_CJ).reg 文件，将所有的 PDS_CJ 替换成 RDS_CJ
   执行即在 HKEY_LOCAL_MACHINE\SOFTWARE\SYBASE\Server 中注册了名为 RDS_CJ 的服务器；

2。编辑 Serv(PDS_CJ).reg 文件，将所有的 PDS_CJ 替换成 RDS_CJ
   执行即在 HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services 中注册了名为 SYBSQL_RDS_CJ 的服务；

3。编辑 Serv(PDS_CJ).reg 文件，将所有的 PDS_CJ 替换成 RDS_CJ
   执行即在
   HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Eventlog\Application 中注册了名为 RDS_CJ 的项目，
   其作用是将 SYBSQL_RDS_CJ 服务的事件信息和错误记录在 Win2000 的事件查看器的应用日志中，

前两步是必需的，第 3 步可以不添加。
以上的注册表文件修改并执行后，不要马上重新启动 Win2000。
请运行 regedit
将 HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SYBSQL_RDS_CJ 中
ImagePath 的数值数据，
由 c:\sybase\ASE-12_0\bin\sqlsrvr.exe -sPDS_CJ -C
该为：
c:\sybase\ASE-12_0\bin\sqlsrvr.exe -sRDS_CJ -C

4。重新启动 Win2000，让注册表中新添加的项目生效。

仅供参考：
WIN NT 4.0 和 WIN2000 的注册表的格式不同，因此不能够将这 2 个平台的注册表文件相互迁移。

5.3.9如何单独在备份机上启动Sybase ASE12.5 (ASE HA)

一．**如何单独在备份机上启动** Sybase ASE12.5

1. 1. **停止** HACMP **的服务进程**
2. 2. **在备机上将卷组激活**
   # varyonvg datavg
3. 3. **修改** standby **地址为主机的** service**地址**
   # smitty tcpip → en2
   修改 10.61.2.102→10.61.1.100
4. 4. **登录到** sybase **用户**
   # su - Sybase
   $
5. 5. **编辑** interfaces **文件**
   将 sdcp_bak_srv **改为** sdcp_pri_srv
6. 6. **启动** ASE
   $ cd /sybase/ASE-12_5/install
   $ startserver –f RUN_sybase_bak
7. 7. **执行如下命令**
   $isql –Usa –P –Ssybase_bak
   1> 1> dbcc takeover
   2> 2> go

二．**恢复到正常状态**

1．1． **行如下命令**
   $isql –Usa –P –Ssybase_bak
   1> dbcc ha_admin(‘’,rollback_failover)
   2> 2> go
2．2． **停止** sybase_bak
   $isql –Usa –P –Ssybase_bak
   1> 1> shutdown
   2> 2> go
3．**将** interfaces **文件中的地址改回为原来的** sdcp_bak_srv
4．**退回到** root **用户**
5．**将地址** 10.61.1.100 **修改回** 10.61.2.102
6．**中断对卷组** datavg **的激活**
   # varyoffvg datavg
7．**启动** HACMP

5.4 数据库恢复以后的工作

5.4.1 如何检查数据库中数据的一致性

数据库一致性检查(dbcc)**提供了一些命令用于检查数据库的逻辑和物理一致性。**Dbcc **主要有两个功能：**

- **使用 checkstorage 或 checktable及 checkdb 在页一级和行一级检查页链及数据指针。**
- **使用 checkstorage, checkalloc, 或 checkverify, tablealloc, 及 indexalloc 检查页分配。**

在下列情况中需要使用 dbcc **命令：**

- **作为数据库日常维护工作的一部分, 数据库内部结构的完整性决定于 sa 或 dbo 定期地运行 dbcc 检查。**
- **在系统报错以后, 确定数据库是否有损坏。**
- **在备份数据库之前, 确保备份的完整性。**
- **如果怀疑数据库有损坏时, 例如, 使用某个表时报出表损坏的信息, 可以使 dbcc 确定数据库中其他表是否也有损坏。**

下面是 dbcc **的简单用法：**

- **dbcc checktable (table_name)**
  检查指定的表, 检查索引和数据页是否正确链接, 索引是否正确排序, 所有指针是否一致, 每页的数据信息是否合理, 页偏移是否合理。
- **dbcc checkdb (database_name)**
  对指定数据库的所有表做和 checktable 一样的检查。
- **dbcc checkalloc (database_name,fix|nofix)**
  检查指定数据库, 是否所有页面被正确分配, 是否被分配的页面没被使用。当使用"fix"选项时，在检查数据库的同时会自动修复有问题的页面。（若数据库数据量很大，则该过程会持续很长时间。）
- **dbcc tablealloc (table_name,fix|nofix)**
  检查指定的表, 是否所有页面被正确分配, 是否被分配的页面没被使用。是 checkalloc 的缩小版本, 对指定的表做完整性检查。当使用"fix"选项时，在检查数据表的同时会自动修复数据表中有问题的页面。

关于上述命令的其它选项及详细使用方法和 checkstorage, checkverify, indexalloc 的详细使用方法, 请参阅 3.11 DBCC 或 Sybase 命令手册。

举例 1**：** Unix **平台检查** pubs2 **数据库的一致性**

- **单用户模式启动** Server**：**
  $SYBASE/install startserver -f RUN_server_name -m
- **vi dbcc_db.sql**

```
use master
go
sp_dboption pubs2,"single user",true
go
use pubs2
go
checkpoint
go
dbcc checkdb(pubs2)
go
dbcc checkalloc(pubs2,fix)
go
dbcc checkcatalog(pubs2)
go
use master
go
sp_dboption pubs2,"single user",false
```

go
use pubs2
go
checkpoint
go
quit
go

- **isql -Usa -Pxxxxxx -SSYBASE <dbcc_db.sql >dbcc_db.out**
- **grep Msg dbcc_db.out**

举例 2**：** Unix **平台检查** pubs2 **数据库中** titles **表的一致性**

- **vi dbcc_table.sql**
  use pubs2
  go
  dbcc checktable(titles)
  go
  dbcc tablealloc(titles)
  go
- **isql -Usa -Pxxxxxx -SSYBASE < dbcc_table.sql > dbcc_table.out**

grep Msg dbcc_table.out

6. SYBASE 培训服务

6.1 SYBASE 培训概述

- **培训方式：**
  - ■ **中心培训：用户可以按照我们在 www.sybase.com.cn 上公布的培训时间表，参加我们在北京、广州和上海培训中心举办的定期标准培训。**
  - ■ **现场培训：我们可以为企业级客户提供位于用户现场的培训服务。**
  - ■ **海外培训：我们也可以应用户的特殊要求，为用户安排海外培训。**
- **培训中心：**

Sybase **公司目前在北京、广州和上海设有自己的培训中心。每个培训中心都拥有完善的培训设施和实验设备，可以保证每个学员都拥有自己独立的实验环境。**

- **培训班课程分类：**
  - ■ **基础级：主要针对使用** Sybase **产品时间较短的用户。**
  - ■ **提高级：适合于有一定** Sybase **产品使用经验，希望进一步系统地提高的用户。**
  - ■ **特殊定制：除上述两种标准形式的培训外，我们还可以根据企业级用户的特殊需求，为其进行专门定制内容的培训。**
- **教材和师资**

  Sybase**的主要数据库培训课程均采用中文教材。**

  Sybase**培训部门的教师都有丰富的教学经验和** Sybase **产品使用经验，所有教师都拥有** Sybase **产品的全球认证和一些其他重要的** IT **认证。**
- **报名方式：**
  - ■ **通过** Sybase **的行业销售或代理商购买培训服务。**
  - ■ **直接向** Sybase **培训部购买培训服务。**
    - ◆ **培训部联系人：赵硕**
    - ◆ **电话：**（010）68568488 – 5106
    - ◆ **传真：**（010）68568489
    - ◆ E-mail: shuo.zhao@sybase.com
- **我们的网址：** **www.sybase.com.cn**

6.2 SYBASE 主要数据库培训班介绍

ASE 基础级

ASE 快速入门班		5 天
课程：	FND200 ASE 快速入门	
本门课程介绍如何使用 ANSI 标准 SQL 和 Sybase 的增强型 Transact_SQL 创建 Sybase 数据库和开发高效率的应用。通过讲解与实践紧密结合的学习方式,让学员掌握如何把 ASE 颇具特色的存储过程，触发器，游标及事务自动恢复等项功能有效地应用于自己的开发实践中去。		

ASE 系统管理班		5 天
课程：	EDB300 ASE 系统管理	
本门课程通过讲解与实验相结合的教学方式，使学员了解 Adaptive Server Enterprise 系统管理的主要任务，掌握全面的系统管理知识。着重介绍如何管理资源（内存，磁盘，CPU）以提供可靠的 Adaptive Server Enterprise 运行环境。此外，还学习监控及排错的基本方法、服务器锁机制，以及系统与用户自定义角色的细节。		

ASE 提高级

ASE 高级系统管理班班		5 天
课程 1：	EDB510 ASE 高级系统管理和排错	
本门课程讲授如何诊断和防止 ASE 潜在故障的发生，以及在常用的管理任务中怎样最有效地使用 ASE 的特性。课程内容还包括怎样更高效的完成关键任务的管理，如何为应用的最终用户改善系统性能，以及在意外故障情况下恢复数据的排错技术。通过与 ASE 内部数据结构有关的实验，该课程提供实际操作机会帮助学员深入的了解 ASE，学到实用的排错技术。		
课程 2：	EDB514 ASE 内部数据结构	
本课提供了实际维护工作中如何有效的确定及解决数据库损坏问题的重要知识。学生将在原有对 ASE 服务器所用的数据结构知识基础上进一步学习相关的内部数据结构知识，以便在一些情况下，学生将有能力独立地解决数据损坏问题。同时相关知识又能增强对服务器及应用方面性能调优的技能。		

ASE 性能调优班		5 天
课程：	EDB512 ASE 性能调优（配置服务器）	
本门课程讲授对 Adaptive Server 进行调优以实现最佳系统性能的方法。课程内容比较深入的描述了 Adaptive Server 的体系结构，同时还包括：高性能数据库的设计，Adaptive Server 的系统资源管理，恰当的锁机制，统计优化，SMP 特征，以及相关的数据结构和并行处理等。		

- Sybase 公司每月在培训中心开设上述培训班，具体时间请查询我们的网站 **www.sybase.com.cn**。
- 如果您想获得上述课程和其它 Sybase 课程的详细介绍，也请查询我们的网站 **www.sybase.com.cn**。

7. 如何获得帮助

7.1 热线电话

凡是购买了 SYBASE **基本级技术支持服务且在服务期内的用户，可以在法定工作日** 09:00 – 18:00 **通过拨打北京热线支持服务中心寻求技术帮助。**

北京技术支持电话：010-68568488 **转 相应产品工程师**

凡是购买了 SYBASE **扩展级技术支持服务且在服务期内的用户，若在非工作时间遇到紧急技术问题，可以拨打各地办事处的** 7x24 **热线电话。**

北京办事处：13901313054　　**上海办事处：**13901666729
成都办事处：13908186755　　**广州办事处：**13802902669

7.2 Sybase 技术文档

凡是购买 Sybase **产品的正式用户，产品包装盒中应该包含一张** Sybase **产品文档光盘。**

7.3 Sybase 网上资源

用户可以通过访问下面的站点浏览 Sybase **产品手册及技术文档：**

中文站点：
Sybase **中国站点：**http://www.sybase.com.cn/
Sybase **产品中文手册：**http://www.sybase.com.cn/cn/content/goto.jsp?resID=185&page=1
Sybase **技术支持：**http://www.sybase.com.cn/cn/content/goto.jsp?resID=5

英文站点：
Sybase **主页：**http://www.sybase.com/
Sybase **技术支持：**http://www.sybase.com/support
Sybase **常见问题：**http://www.sybase.com/detail_list_sort/1,6903,9792,00.html
Sybase **产品手册：**http://www.sybase.com/support/manuals/
国际 SYBASE **用户组：**http://www.isug.com
Sybase **新闻组：**http://www.sybase.com/support/newsgroups

7.4 您的建议

如果您在使用本手册过程中有一些心得，或者您发现本手册有任何错误，或对本手册有些建设性的意见，都请将您的想法 email **或邮寄给我们：**

EMAIL: sybasecn@21cn.com

邮编：100045
地址：北京复外大街 A2 号中化大厦 401 室　技术支持中心收